

10/508802

PATENT
PCT/PTO 23 SEP 2004
450100-04421**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

Applicant: Tomohisa SHIGA

International Application No.: PCT/JP03/03716

International Filing Date: March 26, 2003

For: OPERATION-PROCESSING DEVICE, METHOD FOR
CONSTRUCTING THE SAME, AND OPERATION-
PROCESSING SYSTEM AND METHOD

745 Fifth Avenue
New York, NY 10151

EXPRESS MAIL

Mailing Label Number: EV38541463US

Date of Deposit: September 23, 2004

I hereby certify that this paper or fee is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" Service under 37 CFR 1.10 on the date indicated above and is addressed to Mail Stop PCT, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

Adam Ahmed
(Typed or printed name of person mailing paper or fee)

A. Ahmed
(Signature of person mailing paper or fee)

CLAIM OF PRIORITY UNDER 37 C.F.R. § 1.78(a)(2)

Mail Stop PCT
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

Pursuant to 35 U.S.C. 119, this application is entitled to a claim of priority to Japan
Application No. 2002-088916 and 2002-195123 filed 27 March and 3 July 2002.

Respectfully submitted,

FROMMER LAWRENCE & HAUG LLP
Attorneys for Applicant

By: William S. Frommer
William S. Frommer
Reg. No. 25,506
Tel. (212) 588-0800

BEST AVAILABLE COPY

00219159

Rec'd PCT/PTO 23 SEP 2004
PCT/JP 03/03716 #2

日 本 国 特

JAPAN PATENT

REC 許 7 A 府 2003

OFFICE

WIPO

PCT

26.03.03

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出 願 年 月 日

Date of Application:

2002年 7月 3日

出 願 番 号

Application Number:

特願2002-195123

[ST.10/C]:

[JP2002-195123]

出 願 人

Applicant(s):

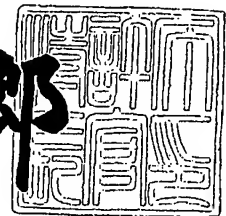
ソニー株式会社

PRIORITY DOCUMENT
SUBMITTED OR TRANSMITTED IN
COMPLIANCE WITH
RULE 17.1(a) OR (b)

2003年 2月14日

特 許 庁 長 官
Commissioner,
Japan Patent Office

太田 信一郎



出証番号 出証特2003-3007234

BEST AVAILABLE COPY

【書類名】 特許願

【整理番号】 0290181903

【提出日】 平成14年 7月 3日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 15/76
G06F 15/78
H03K 19/173

【発明者】

【住所又は居所】 東京都品川区北品川6丁目7番35号 ソニー株式会社
内

【氏名】 志賀 知久

【特許出願人】

【識別番号】 000002185

【氏名又は名称】 ソニー株式会社

【代理人】

【識別番号】 100090376

【弁理士】

【氏名又は名称】 山口 邦夫

【電話番号】 03-3291-6251

【選任した代理人】

【識別番号】 100095496

【弁理士】

【氏名又は名称】 佐々木 榮二

【電話番号】 03-3291-6251

【手数料の表示】

【予納台帳番号】 007548

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9709004

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 データ処理システム、データ処理装置及びデータ処理方法

【特許請求の範囲】

【請求項1】 一方で、所定のプログラム言語に基づいて目的の演算処理を実行するための命令を編集してプログラムを作成し、他方で、当該プログラムと複数のレジスタとを使用してデータを処理するシステムであって、

前記レジスタを使用する頻度に基づいて当該レジスタを指定する命令ビット数を圧縮すると共に、当該プログラムの命令構造の中にレジスタ種類を記述して命令長の異なる圧縮プログラムを作成するプログラム作成装置と、

前記プログラム作成装置で作成された圧縮プログラムを取得してレジスタ種類を解読し、前記レジスタ種類に基づいて当該レジスタを指定する命令ビット数を復元し、所定長さの命令に基づいて複数のレジスタを指定して任意の演算を実行するデータ処理装置とを備えることを特徴とするデータ処理システム。

【請求項2】 前記データ処理装置は、

複数のレジスタと、

前記レジスタを指定するための圧縮プログラムを記憶する記憶手段と、

前記記憶手段から圧縮プログラムを読み出してレジスタ種類を解読し、当該レジスタ種類に基づいて前記レジスタを指定するための命令ビット数を復元する命令解読復元手段と、

前記命令解読復元手段によって復元された所定の長さの命令に基づいて前記レジスタを指定して任意の演算を実行する命令実行演算手段とを有することを特徴とする請求項1に記載のデータ処理システム。

【請求項3】 前記レジスタ種類は、

N個のレジスタを使用する場合であって、

前記N個のレジスタに第1番から第N番のシリアル番号を付与したとき、

前記第1番から第K番のグループのレジスタを使用頻度が高い部類に、

前記第K+1番から第N番のグループのレジスタを使用頻度が低い部類に分けることを特徴とする請求項1に記載のデータ処理システム。

【請求項4】 レジスタを使用する頻度に基づいて当該レジスタを指定する

命令ビット数が予め圧縮されると共に、当該プログラムの命令構造の中にレジスタ種類が記述された命令長の異なる圧縮プログラムに基づいてデータを処理する装置であって、

複数のレジスタと、

前記レジスタを指定するための圧縮プログラムを記憶する記憶手段と、

前記記憶手段から圧縮プログラムを読み出してレジスタ種類を解読し、当該レジスタ種類に基づいて前記レジスタを指定するための命令ビット数を復元する命令解読復元手段と、

前記命令解読復元手段によって復元された所定長さの命令に基づいて前記レジスタを指定して任意の演算を実行する命令実行演算手段とを備えることを特徴とするデータ処理装置。

【請求項5】 前記レジスタ種類は、

N個のレジスタを使用する場合であって、

前記N個のレジスタに第1番から第N番のシリアル番号を付与したとき、

前記第1番から第K番のグループのレジスタを使用頻度が高い部類に、前記第K+1番から第N番のグループのレジスタを使用頻度が低い部類に分けられることを特徴とする請求項4に記載のデータ処理装置。

【請求項6】 プログラム作成系で所定のプログラム言語に基づいて目的の演算処理を実行するための命令を編集してプログラムを作成し、プログラム実行系で当該プログラムと複数のレジスタとを使用してデータを処理する方法であって、

前記プログラム作成系では、

前記レジスタを使用する頻度に基づいて当該レジスタを指定する命令ビット数を圧縮すると共に、当該プログラムの命令構造の中にレジスタ種類を記述して命令長の異なる圧縮プログラムを作成し、

前記プログラム実行系では、

前記プログラム作成系で作成された圧縮プログラムを取得してレジスタ種類を解読し、

解読された前記レジスタ種類に基づいて当該レジスタを指定する命令ビット数

を復元し、

復元された所定長さの前記命令に基づいて複数のレジスタを指定して任意の演算を実行することを特徴とするデータ処理方法。

【請求項7】 前記レジスタ種類は、

N個のレジスタを使用する場合であって、

前記N個のレジスタ第1番から第N番のシリアル番号を付与したとき、

前記第1番から第K番のグループのレジスタを使用頻度が高い部類に、

前記第K+1番から第N番のグループのレジスタを使用頻度が低い部類に分けることを特徴とする請求項6に記載のデータ処理方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、システムプログラムに基づいて各種データ処理をする中央演算装置（CPU）やマイクロプロセッサユニット（MPU）等、また、プログラム可能な論理演算素子（PLD）や、これらの組み込み電子機器等に適用して好適なデータ処理システム、データ処理装置及びデータ処理方法に関する。

【0002】

詳しくは、複数のレジスタを指定して任意の演算を実行するデータ処理装置を備え、プログラム作成系で作成された圧縮プログラムを取得してレジスタ種類を解読し、このレジスタ種類に基づいて当該レジスタを指定する命令ビット数を復元し、所定の命令長の命令構造を有するプログラムを復元するようにして、プログラムデータを格納するROM等のメモリ容量を低減できるようにすると共に、メモリセルや論理演算素子から成るPLDによりプロセッサを構築する場合に、ROMとして機能させるメモリセルの占有率を低減できるようにしたものである。

【0003】

【従来の技術】

近年、携帯端末装置や、電子カード、情報処理装置等の各種電子機器にCPU（中央演算処理装置）を含むマイクロプロセッサが使用される場合が多くなって

きた。この種のプロセッサには命令実行演算部の他に命令実行プログラムを格納するための読み出し専用メモリ（以下、ROMという）や、命令実行演算処理に使用される多くのレジスタ等が実装されている。

【0004】

従来方式のマイクロプロセッサによれば、任意の電子機器に当該プロセッサを組み込んで用いる場合、ある動作を行う命令はその動作と、その動作を行うための命令とが一对一に対応していた。つまり、使用頻度が高いレジスタも、使用頻度が低いレジスタも、当該レジスタを指定するための命令ビット数を同等にして一律の長さの命令により作成されたプログラムが使用される場合が多い。このことで、固定長の命令がROMに格納されて使用される。

【0005】

他方で、半導体集積回路技術の発展により、多大な数のレジスタをプロセッサ（以下データ処理装置ともいう）内に実装することが可能になってきた。この場合、レジスタを特定するための命令ビット数も益々多く必要になる。例えば、1024個のレジスタが実装される場合、1024個のレジスタ中でその1つを特定するためには、命令ビットとして10ビットが必要となる。しかし、実際のプログラムでは全てのレジスタへのアクセス頻度は一様ではなく、アクセス頻度に差がある。頻繁にアクセスされるレジスタの番号は一般的にコンパイラによって決定される。

【0006】

【発明が解決しようとする課題】

ところで、従来方式のデータ処理装置によれば、命令実行プログラムを格納するROMを実装するに当たり以下のような問題がある。

【0007】

① 命令実行プログラムのコード全体を眺めてみると、レジスタを指定するための例えば、10ビットの命令ビット内、この10ビット全体が一様に使用されることが少ない。従って、命令実行プログラムを格納するメモリ中（例えばROMやフラッシュメモリ）に無駄なビットが多く存在してしまう。これにより、全てのレジスタを単一の命令ビット数で表現する方法は効率良くROMを使用する

ことに関して妨げとなる。

【0008】

② また、メモリセルや論理演算素子から成るプログラマブル・ロジック・デバイス (Programmable Logic Device; PLD) によりマイクロプロセッサ等を構築しようとした場合に、同一半導体チップ上に命令実行演算部を配置し、その周辺部にレジスタアレイや、ROM等を配置する方法が考えられる。この場合に、プロセッサの多機能化の要求から命令実行プログラムが多くなると、このプログラムを格納するROMのメモリ容量が多く必要になる。従って、メモリセルがROM構築に占有されてしまい、多くのメモリセルをレジスタに割り当てることが困難になる。

【0009】

そこで、この発明はこのような従来の課題を解決したものであって、レジスタの使用頻度に応じて命令の長さを可変できるようにすると共に、頻繁にアクセスするレジスタには短い長さの命令をセットできるようにし、プログラムデータを格納するROM等のメモリ容量を低減できるようにしたデータ処理システム、データ処理装置及びデータ処理方法を提供することを目的とする。

【0010】

【課題を解決するための手段】

上述した課題は、一方で、所定のプログラム言語に基づいて目的の演算処理を実行するための命令を編集してプログラムを作成し、他方で、当該プログラムと複数のレジスタとを使用してデータを処理するシステムであって、レジスタを使用する頻度に基づいて当該レジスタを指定する命令ビット数を圧縮すると共に、当該プログラムの命令構造の中にレジスタ種類を記述して命令長の異なる圧縮プログラムを作成するプログラム作成装置と、このプログラム作成装置で作成された圧縮プログラムを取得してレジスタ種類を解読し、レジスタ種類に基づいて当該レジスタを指定する命令ビット数を復元し、所定長さの命令に基づいて複数のレジスタを指定して任意の演算を実行するデータ処理装置とを備えることを特徴とするデータ処理システムによって解決される。

【0011】

本発明に係るデータ処理システムによれば、一方で、所定のプログラム言語に基づいて目的の演算処理を実行するための命令を編集してプログラムを作成し、他方で、当該プログラムと複数のレジスタとを使用してデータを処理する場合に、プログラム作成装置ではレジスタを使用する頻度に基づいて当該レジスタを指定する命令ビット数を圧縮すると共に、当該プログラムの命令構造の中にレジスタ種類を記述して命令長の異なる圧縮プログラムが作成される。

【 0 0 1 2 】

データ処理装置では、プログラム作成装置で作成された圧縮プログラムを取得してレジスタ種類を解釈し、このレジスタ種類に基づいて当該レジスタを指定する命令ビット数を復元し、所定長さの命令に基づいて複数のレジスタを指定して任意の演算が実行される。

【 0 0 1 3 】

従って、プログラム作成系ではレジスタの使用頻度に応じて命令の長さを可変できるので、頻繁にアクセスするレジスタに短い長さの命令をセットすることができる。これにより、プログラム実行系ではROM等に圧縮した命令をセットすることができ、プログラムデータを格納するROM等のメモリ容量を低減することができる。また、メモリセルや論理演算素子から成るPLDによりプロセッサを構築する場合に、ROMとして機能させるメモリセルの占有率を低減することができ、その分のメモリセルをレジスタに多く割り当てることができる。

【 0 0 1 4 】

本発明に係るデータ処理装置は、レジスタを使用する頻度に基づいて当該レジスタを指定する命令ビット数が予め圧縮されると共に、当該プログラムの命令構造の中にレジスタ種類が記述された命令長の異なる圧縮プログラムに基づいてデータを処理する装置であって、複数のレジスタと、レジスタを指定するための圧縮プログラムを記憶する記憶手段と、この記憶手段から圧縮プログラムを読み出してレジスタ種類を解釈し、当該レジスタ種類に基づいてレジスタを指定するための命令ビット数を復元する命令解釈復元手段と、この命令解釈復元手段によって復元された所定長さの命令に基づいてレジスタを指定して任意の演算を実行する命令実行演算手段とを備えることを特徴とするものである。

【0015】

本発明に係るデータ処理装置によれば、レジスタを使用する頻度に基づいて当該レジスタを指定する命令ビット数が予め圧縮されると共に、当該プログラムの命令構造の中にレジスタ種類が記述された命令長の異なる圧縮プログラムに基づいてデータを処理する場合に、記憶手段には複数のレジスタ中から当該レジスタを指定するための圧縮プログラムが記憶される。命令解読復元手段では、この記憶手段から圧縮プログラムを読み出してレジスタ種類が解読され、このレジスタ種類に基づいて当該レジスタを指定するための命令ビット数が復元される。これを前提にして、命令実行演算手段では命令解読復元手段によって復元された所定の命令長のプログラムに基づいてレジスタを指定して任意の演算を実行するようになされる。

【0016】

従って、レジスタの使用頻度に応じて可変された命令の長さの圧縮プログラムであって、頻繁にアクセスするレジスタには短い長さの命令がセットされた、圧縮プログラムデータを記憶手段に格納することができるので、そのメモリ容量を低減することができる。これにより、メモリセルや論理演算素子から成るPLDによりプロセッサを構築する場合に、ROMとして機能させるメモリセルの占有率を低減することができ、その分のメモリセルをレジスタに多く割り当てることができる。

【0017】

本発明に係るデータ処理方法はプログラム作成系で所定のプログラム言語に基づいて目的の演算処理を実行するための命令を編集してプログラムを作成し、プログラム実行系で当該プログラムと複数のレジスタとを使用してデータを処理する方法であって、プログラム作成系では、レジスタを使用する頻度に基づいて当該レジスタを指定する命令ビット数を圧縮すると共に、当該プログラムの命令構造の中にレジスタ種類を記述して命令長の異なる圧縮プログラムを作成し、プログラム実行系では、プログラム作成系で作成された圧縮プログラムを取得してレジスタ種類を解読し、ここで解読されたレジスタ種類に基づいて当該レジスタを指定する命令ビット数を復元し、ここで復元された所定長さの命令に基づいて複

数のレジスタを指定して任意の演算を実行することを特徴とするものである。

【0018】

本発明に係るデータ処理方法によれば、プログラム作成系で所定のプログラム言語に基づいて目的の演算処理を実行するための命令を編集してプログラムを作成し、プログラム実行系で当該プログラムと複数のレジスタとを使用してデータを処理する場合に、プログラム作成系ではレジスタの使用頻度に応じて命令の長さを可変できるので、頻繁にアクセスするレジスタに短い長さの命令をセットすることができる。

【0019】

従って、プログラム実行系ではROM等に圧縮した命令をセットすることができ、プログラムデータを格納するROM等のメモリ容量を低減することができる。また、メモリセルや論理演算素子から成るPLDによりプロセッサを構築する場合に、ROMとして機能させるメモリセルの占有率を低減することができ、その分のメモリセルをレジスタに多く割り当てることができる。

【0020】

【発明の実施の形態】

続いて、この発明に係るデータ処理システム、データ処理装置及びデータ処理方法の一実施の形態について、図面を参照しながら説明をする。

【0021】

(1) 実施形態

図1は本発明に係る実施形態としてのデータ処理システム100の構成例を示すブロック図である。

この実施形態では複数のレジスタを指定して任意の演算を実行するデータ処理装置を備え、プログラム作成系で作成された圧縮プログラムを取得してレジスタ種類を解読し、このレジスタ種類に基づいて当該レジスタを指定する命令ビット数を復元し、所定の命令長の命令構造を有するプログラムを復元するようにして、プログラムデータを格納するROM等のメモリ容量を低減できるようにすると共に、メモリセルや論理演算素子から成るPLDによりプロセッサを構築する場合に、ROMとして機能させるメモリセルの占有率を低減できるようにしたもの

である。

【0022】

また、プログラム作成系ではレジスタの使用頻度に応じて命令の長さを可変できるようにすると共に、頻繁にアクセスするレジスタには短い長さの命令をセットできるようにする。

【0023】

図1に示すデータ処理システム10は、一方で、所定のプログラム言語に基づいて目的の演算処理を実行するための命令を編集してプログラムを作成し、他方で、当該プログラムと複数のレジスタとを使用してデータを処理するシステムである。データ処理システム10ではプログラム作成系Iを成すプログラム作成装置200が準備される。新規に設計製造されるデータ処理装置100を動作させるためのプログラムを作成するためである。データ処理装置100はプログラム実行系IIを構成し、当該装置100内には命令実行演算手段や、記憶手段、複数のレジスタ等が実装される。

【0024】

プログラム作成装置200では、データ処理装置100でレジスタを使用する頻度に基づいて当該レジスタを指定するための命令ビット数を圧縮すると共に、当該プログラムの命令構造の中にレジスタ種類を記述して命令長の異なる圧縮プログラムAPを作成するようになされる。データ処理装置100に実装されるプログラム格納用の記憶手段のメモリ容量を削減するためである。

【0025】

プログラム作成装置200は例えば、データベース21、キーボード22、マウス23、表示装置24及び制御装置25を有している。データベース21にはデータ処理装置100のプログラム作成に必要なデータが格納される。例えば、C言語によるプログラムの記述に必要な「Global変数宣言」、「関数宣言」、「Local変数宣言」、「代入」、「加算」、「比較」及び「分岐」が格納される。データベース21には制御装置25が接続されており、この制御装置25にはキーボード22、マウス23及び表示装置24が接続されている。

【0026】

プログラム作成装置200では表示装置24にC言語によるプログラム記述画面を表示して、キーボード22及びマウス23を使用してプログラムが作成される。例えば、新規な設計製造に係るデータ処理装置100がN個のレジスタを使用する場合であって、N個のレジスタに第1番から第N番のシリアル番号を付与したとき、第1番から第K番のグループのレジスタを使用頻度が高い部類として「Local変数宣言」がなされ、第K+1番から第N番のグループのレジスタを使用頻度が低い部類として「Global変数宣言」がなされる。

【0027】

これらの宣言はキーボード22や、マウス23を使用して指定され、これはレジスタ種類を2つに分類して使用頻度が高いレジスタは短い命令ビット数で命令セットし、使用頻度が低いレジスタは長い命令ビット数で命令セットするためである。データ処理装置100においてレジスタの数が例えば、4千～8千個程度になると、これを指定する命令ビット数が12乃至13ビット必要になる。

【0028】

制御装置25ではレジスタの使用頻度に応じて命令の長さを可変するようになされる。頻繁にアクセスするレジスタは短い長さの命令をセットするためである。使用頻度が高いレジスタは短い命令ビット数が割り当てられ、使用頻度が低いレジスタは長い命令ビット数が割り当てられる。

【0029】

データ処理装置100ではこのプログラム作成装置200で作成された圧縮プログラムAPを取得してレジスタ種類を解読し、このレジスタ種類に基づいて当該レジスタを指定する命令ビット数を復元し、所定長さの命令に基づいて複数のレジスタを指定して任意の演算を実行するようになされる。

【0030】

データ処理装置100は例えば、命令解読復元手段3、記憶手段4、レジスタアレイ11及び命令実行演算手段50を有している。レジスタアレイ11は複数のレジスタを集合したものである。

【0031】

記憶手段4にはレジスタアレイ11の中から該当レジスタを指定するための圧

縮プログラムAPが格納される。圧縮プログラムAPはプログラム作成装置200で作成されたものが使用される。例えば、圧縮プログラムAPはデータ処理装置100で構築されたプログラム格納用の記憶手段4に、ROMライタ等を使用して書き込まれる。

【0032】

これはデータ処理装置100に関して、複数のメモリセルや算術論理素子により構成されるプログラマブル・ロジック・デバイス(Programmable Logic Device; PLD)からプロセッサを構築する場合があるからである。ROMとして機能させるメモリセルの占有率を低減することができる。もちろん、プログラム格納用の記憶手段4を演算処理装置とは別個に製造し、個々の記憶手段4に圧縮プログラムAPを格納してから同一基板上に実装する方法であってもよい。記憶手段4として読み出し専用のメモリ(ROM)や、EEPROM(フラッシュメモリ)が使用されるからである。

【0033】

記憶手段4には命令解読復元手段3が接続されており、この記憶手段4から圧縮プログラムAPを読み出してレジスタ種類を解読し、このレジスタ種類に基づいて当該レジスタを指定するための命令ビット数を復元するようになされる。命令長を揃え、この命令に基づいて複数のレジスタを指定するためである。

【0034】

命令解読復元手段3には命令実行演算手段50が接続されており、この命令実行演算手段50にはレジスタアレイ11が接続されている。命令実行演算手段50では命令解読復元手段3によって復元された所定の命令長のプログラムに基づいてレジスタアレイ11内で該当レジスタを指定して任意の演算を実行するようになされる。

【0035】

続いて、本発明に係るデータ処理方法について、当該データ処理システム10における処理例について説明をする。図2はデータ処理システム10における処理例を示すフローチャートである。

【0036】

このシステム 1 0 ではプログラム作成系 I で所定のプログラム言語に基づいて目的の演算処理を実行するための命令を編集してプログラムを作成し、プログラム実行系 II で当該プログラムと複数のレジスタとを使用してデータを処理する場合を前提とする。このデータ処理装置 1 0 0 が N 個のレジスタを使用する場合であって、N 個のレジスタに第 1 番から第 N 番のシリアル番号が付与される場合を例にとる。

【 0 0 3 7 】

これを処理条件にして、プログラム作成系では図 2 A にフローチャートのステップ A 1 で所定のプログラム言語に基づいて目的の演算処理を実行するための命令を編集する。そして、ステップ A 2 でレジスタを使用する頻度に基づいて当該レジスタを指定する命令ビット数を圧縮し、命令長を短くする。例えば、第 K + 1 番から第 N 番のグループのレジスタを指定する命令ビット数を n ビットとし、第 1 番から第 K 番のグループのレジスタを指定する命令ビット数を m ビットとしたとき、例えば、 $n - m = 8$ ビットとなるように、第 1 番から第 K 番のグループのレジスタを指定する命令ビット数が圧縮される。

【 0 0 3 8 】

その後、ステップ A 3 で当該プログラムの命令構造の中にレジスタ種類を記述する。例えば、被数及び加数を保持する 1 組のレジスタ、これを「レジスタ番号 1」のレジスタの種類を「レジスタ種類 1」、及び「レジスタ番号 2」のレジスタの種類を「レジスタ種類 2」としたとき、第 1 番から第 K 番のグループのレジスタに関して使用頻度が高い場合は「レジスタ種類 1」及び「レジスタ種類 2」にコード「0」が記述される。また、第 K + 1 番から第 N 番のグループのレジスタに関して使用頻度が低い場合は「レジスタ種類 1」及び「レジスタ種類 2」にコード「1」が記述される。

【 0 0 3 9 】

そして、ステップ A 4 で命令長の異なる圧縮プログラム A P を作成する。この圧縮プログラム A P において、第 1 番から第 K 番のグループのレジスタを指定する命令ビット数に関しては m ビットであり、第 K + 1 番から第 N 番のグループのレジスタを指定する命令ビット数に関しては n ビットである。上述の例で第 1 番

から第K番のグループのレジスタ指定を含む命令形態では、第K+1番から第N番のグループのレジスタ指定を含む命令形態に比べて命令長が16ビット短くなる。

【0040】

一方、プログラム実行系ではプログラム作成系で作成された圧縮プログラムAPを図2Bに示すフローチャートのステップB1で取得する。例えば、圧縮プログラムAPはデータ処理装置100で構築されたプログラム格納用の記憶手段4に、ROMライタ等を使用して書き込まれる。この圧縮プログラムAPでは使用頻度が高いレジスタは短い命令ビット数=mビットで命令セットされ、使用頻度が低いレジスタは長い命令ビット数=nビットで命令セットされている。

【0041】

そして、ステップB2で命令を実行するかを判断する。この際の判断は周知技術によりなされる。命令を実行する場合はステップB3でレジスタ種類を解読する。例えば、「レジスタ種類1」及び「レジスタ種類2」に関してコード「0」から使用頻度が高いレジスタとして第1番から第K番のグループのレジスタ番号が解読され、「レジスタ種類1」及び「レジスタ種類2」に関してコード「1」から使用頻度が低いレジスタとして第K+1番から第N番のグループのレジスタ番号が解読される。

【0042】

そして、解読されたレジスタ種類に基づいてステップB4で当該レジスタを指定する命令ビット数を復元する。例えば、第1番から第K番のグループのレジスタの命令ビット数=mビットの上位、この例で上位8ビットに「0」が付加される。第1番から第K番のグループのレジスタの命令ビット数が、第K+1番から第N番のグループのレジスタの命令ビット数と同様にしてnビットに揃えられる。

【0043】

ここで復元された所定長さの命令に基づいてステップB5で複数のレジスタを指定して任意の演算を実行する。その後、ステップB6で演算処理を終了するかを判断する。演算処理を終了しない場合はステップB2に戻って命令を実行する

かを判断して演算処理を継続する。演算処理を終了する場合は電源オフ情報等を検出して当該演算処理を終了する。

【0044】

このように、本発明に係る実施形態としてのデータ処理システム10によれば、一方で、所定のプログラム言語に基づいて目的の演算処理を実行するための命令を編集してプログラムを作成し、他方で、当該プログラムと複数のレジスタとを使用してデータを処理する場合に、プログラム作成装置200ではレジスタを使用する頻度に基づいて当該レジスタを指定する命令ビット数を圧縮すると共に、当該プログラムの命令構造の中にレジスタ種類を記述して命令長の異なる圧縮プログラムAPが作成される。

【0045】

データ処理装置100では、プログラム作成装置200で作成された圧縮プログラムAPを取得してレジスタ種類を解読し、このレジスタ種類に基づいて当該レジスタを指定する命令ビット数を復元し、所定長さの命令に基づいて複数のレジスタを指定して任意の演算が実行される。

【0046】

従って、プログラム作成系Iではレジスタの使用頻度に応じて命令の長さを可変できるので、頻繁にアクセスするレジスタに短い長さの命令をセットすることができる。これにより、プログラム実行系IIではROM等の記憶手段4に圧縮した命令をセットすることができ、プログラムデータを格納する記憶手段4のメモリ容量を低減することができる。また、メモリセルや論理演算素子から成るPLDによりプロセッサを構築する場合に、ROMとして機能させるメモリセルの占有率を低減することができ、その分のメモリセルをレジスタに多く割り当てることができるようになる。

【0047】

(2) 実施例

図3は本発明に係る実施例としてのマイクロプロセッサ101の構成例を示すブロック図である。

この実施例ではデータ処理装置100に外部メモリ2を接続してマイクロプロ

セッサ101を構成し、複数のレジスタを指定して任意の演算を実行する。そのために、プログラム作成系Iで作成された機械語の命令の圧縮プログラムAPを取得してレジスタ種類を解読し、このレジスタ種類に基づいて当該レジスタを指定する命令ビット数を復元し、所定の命令長の命令構造を有するプログラムを復元するようにした。そうすることでプログラムデータを格納するROM等のメモリ容量を低減できるようにしたものである。

【0048】

図3に示すマイクロプロセッサ101は、レジスタを使用する頻度に基づいて当該レジスタを指定する命令ビット数が予め圧縮されると共に、当該プログラムの命令構造の中にレジスタ種類が記述された命令長の異なる圧縮プログラムAPに基づいてデータを処理する装置である。このプロセッサ101ではプログラム作成装置200で作成された圧縮プログラムAPを取得してレジスタ種類を解読し、このレジスタ種類に基づいて当該レジスタを指定する命令ビット数を復元し、所定長さの命令に基づいて複数のレジスタを指定して任意の演算を実行するようになされる。

【0049】

マイクロプロセッサ101は例えば、レジスタアレイ11、命令ビット復元デコーダ13、ROM14、及び命令実行演算手段50を有している。レジスタアレイ11は複数のレジスタを集合したものである。レジスタアレイ11には例えば、8192個×32bitのレジスタ r_i ($i=0\sim 8191$) が設けられる。各々のレジスタ r_i は書込みアドレス A_w 及び書込み制御信号 S_w に基づいて任意の値を保持し、及び、読出しアドレス A_r に基づいて被数 X や加数 Y 等の値を出力するようになされる。

【0050】

このマイクロプロセッサ101は記憶手段の一例となるプログラム格納用のROM14が実装されており、レジスタアレイ11の中から該当レジスタ r_i を指定するための圧縮プログラムAPが格納される。圧縮プログラムAPは機械語の命令(Instruction)構造を有しており、プログラム作成装置200で作成されたものが使用される。例えば、圧縮プログラムAPはROMライター等を使用して

ROM14に書き込まれる。命令実行時、ROM14は例えば、プログラムカウンタ54からのカウント出力信号S5に基づいて圧縮プログラムAPを出力するようになされる。

【0051】

ROM14には命令ビット復元デコーダ13が接続されており、このROM14から機械語の命令の圧縮プログラムAPを読み出して命令制御信号S4、命令信号S9及び各引数信号S10を発生するようになされる。命令信号S9にはload命令、add命令、cmp命令、jump命令が含まれる。各引数信号S10にはアクセス方法#1、アクセス方法#2、「レジスタ種類1」、「レジスタ種類2」、レジスタ番号r0、r1・・・等、フラグ状態(flag condition)及びジャンプアドレス等が含まれる。

【0052】

この圧縮プログラムAPには、レジスタ相対メモリアドレッシング処理を実行するための演算命令を含んでいる。この処理では演算命令に基づいて一のレジスタを選択し、ここで選択されたレジスタが保持する値によって外部メモリ2を選択するようになされる。この処理は例えば、アクセス方法#1によって実行される。

【0053】

この例で命令ビット復元デコーダ13は、「レジスタ種類1」及び「レジスタ種類2」を解読し、この「レジスタ種類1」及び「レジスタ種類2」に基づいて当該レジスタriを指定するための命令ビット数=nビットを復元するようになされる。命令長を圧縮前のビット数に揃え、この命令に基づいて複数のレジスタri等を指定するためである。上述の命令制御信号S4は命令読出しステートマシーン52に出力される。

【0054】

上述のレジスタアレイ11及び命令ビット復元デコーダ13には命令実行演算手段50が接続されている。命令実行演算手段50では命令ビット復元デコーダ13によって復元された所定の命令長のプログラムに基づいてレジスタアレイ11内で該当レジスタriを指定して任意の演算を実行するようになされる。

【0055】

命令実行演算手段50は、算術論理演算ユニット (Arithmetic and Logic Unit: 以下でALUという) 12、実行ステートマシーン51、命令読出しステートマシーン52、セクタ53、プログラムカウンタ (PC) 54、+1インクリメンタ55、入力用のセクタ59、ラッチ回路58、510、511を有しており、レジスタ相対メモリアドレス処理を実行するようになされる。

【0056】

レジスタアレイ11にはデータ信号線L20が接続されており、この信号線L20を通じてALU12が接続されている。ALU12ではレジスタアレイ11の中で指定されたレジスタから読み出されたXやY等の値を演算するようになされる。演算結果の値はZである。演算種目は足し算、かけ算、引き算、わり算等である。演算種目は実行ステートマシーン51から出力されるALU制御信号S35に基づいて設定される。データ信号線L20には、ALU12の他にラッチ回路58、511、59等が接続されている。データ信号線L20にはDATA、被数X値、加数Y値等が伝送される。

【0057】

命令ビット復元デコーダ13には実行ステートマシーン51及び命令読出しステートマシーン52が接続されており、命令ビット復元デコーダ13によって解読された演算命令を実行するためにレジスタアレイ11及びALU12を制御するようになされる。

【0058】

命令読出しステートマシーン52では命令ビット復元デコーダ13から出力される命令制御信号S4に基づいてプログラムカウンタ54及び実行ステートマシーン51を制御する。例えば、当該マシーン52は命令ビット復元デコーダ13から命令信号S9及び各引数信号S10が実行ステートマシーン51へ出力されると共に命令実行開始信号S29を出力する。

【0059】

実行ステートマシーン51にはレジスタアレイ11、ALU12、ラッチ回路

58、510、511及びセクタ59が接続されている。当該マシーン51では命令実行開始信号S29に基づいて命令の実行を開始する。例えば、データの書込み時には、書込み制御信号Swがレジスタアレイ11に出力され、セクタ59には選択制御信号S24が出力される。データの読出し時には、レジスタアレイ11に読出しアドレスArが出力される。

【0060】

演算時には、ラッチ制御信号S34がラッチ回路58に出力され、ラッチ回路510にはラッチ制御信号S38が出力される。当該プロセッサ外部には外部制御信号S16が出力される。命令の実行が終了すると、実行ステートマシーン51は命令読み出しステートマシーン52へ実行終了信号S26を出力し、プログラムカウンタ54の値を進めるようになされる。

【0061】

この実行ステートマシーン51及び命令読み出しステートマシーン52にはセクタ53が接続されており、選択制御信号S28に基づいてインクリメント出力信号S7又は分岐制御信号S27のいずれか一方を選択し、これをセクタ出力としてプログラムカウンタ54に出力するようになされる。選択制御信号S28は実行ステートマシーン51から供給される。インクリメント出力信号S7はインクリメンタ55からセクタ53へ出力される。

【0062】

プログラムカウンタ54ではカウント制御信号S30に基づいてROM14から圧縮プログラムAPを読み出す場所が指定される。+1インクリメンタ55はプログラムカウンタ54のカウント出力信号S5を「+1」してインクリメントするようになされる。カウント制御信号S30は命令解読ステートマシーン52から供給される。このカウント出力信号S5は+1インクリメンタ55の他にROM14に出力される。

【0063】

セクタ59はデータバス19A、レジスタアレイ11及びALU12に接続されており、データバス19Aから取り込んだデータ(DATA)、レジスタアレイ11から出力される被数X値(加数Y値)又はALU12から出力される演

算結果値Zのいずれかを選択制御信号S24に基づいて入力制御するようになされる。

【0064】

ラッチ回路58はレジスタアレイ11の読出しポートとALU12の間に接続されており、ラッチ制御信号S34に基づいてレジスタriの出力値Xをラッチするようになされる。ラッチ回路510はALU12の比較出力部等に接続されており、ラッチ制御信号S38に基づいて一致検出信号S22をラッチして、フラグ状態(flag condition)信号S23を出力するようになされる。ラッチ回路511はレジスタアレイ11の読出しポートとアドレスバス19Bとの間に接続されており、ラッチ制御信号S17に基づいて外部アドレス(address)をラッチするようになされる。

【0065】

なお、実行された命令によってジャンプ(命令分岐)が発生した場合は、ジャンプ先のアドレスを示す分岐制御信号S27を実行ステートマシン51からセレクタ53へ出力される。セレクタ53では選択制御信号S28に基づいてその分岐制御信号S27を選択し、この分岐制御信号S27をプログラムカウンタ54へ書き込むようになされる。

【0066】

また、実行ステートマシン51、セレクタ59、データ信号線L20及びラッチ回路511にはI/Oインタフェース60を通じて外部メモリ2が接続されている。レジスタ相対メモリアドレスリング処理に基づいてALU12を動作させるためである。I/Oインタフェース60と外部メモリ2との間はデータバス19A、アドレスバス19B及びコントロールバス19Cによって接続され、データバス19Aによってデータが転送され、アドレスバス19Bによってアドレスが転送され、コントロールバス19Cによって外部制御信号S16が外部メモリ2へ転送される。外部メモリ2を制御するためである。外部メモリ2には例えば、512Mバイト×32ビットのRAM(随時書き込み読み出し可能なメモリ)が使用される。

【0067】

図4はレジスタアレイ11の内部構成例を示すブロック図である。図4に示すレジスタアレイ11によれば、例えば、8192個の32bitのレジスタ r_i ($i=0\sim 8191$)が備えられ、各々のレジスタ r_i の入力には書き込みポート15が接続されている。1bitのレジスタはD型のフリップ・フロップ回路等から構成される。

【0068】

書き込みポート15は図3に示したセクタ59に接続されており、書き込み制御信号 S_w 及び書き込みアドレス A_w に基づいて、データバス19Aから取り込んだデータ(DATA)、レジスタアレイ11から出力される被数 X 値(加数 Y 値)又はALU12から出力される演算結果値 Z のいずれかをレジスタ $r_0\sim r_i$ に書き込むようになされる。書き込みポート15は実行ステートマシーン51に接続され、書き込みアドレス A_w を供給するようになされる。

【0069】

各々のレジスタ r_i の出力には読出しポート16が接続されている。読出しポート16はデータ信号線L20を通じて図3に示したALU12、ラッチ回路58、511、セクタ59等に接続されており、読出しアドレス A_r に基づいて指定されたレジスタ r_i からデータ(DATA)を読み出すようになされる。読出しポート16は実行ステートマシーン51に接続され、読出しアドレス A_r を供給するようになされる。

【0070】

次に、ROM14にセットされる命令の構造例について説明する。図5A～Eはマイクロプロセッサ101で取り扱う命令の構造例を示すフォーマットである。図6A～Dは命令構造における記述内容例を示す表図である。

【0071】

このROM14にセットされる命令形態は、図5A～Eに示すように、#F1～#F5の5種類である。マイクロプロセッサ101では命令形態#F2～#F4の命令を命令形態#F1の命令の形態に復元して取り扱われる。これらの命令形態#F1～#F5で各々の命令はload、add及びcmp命令と、jump命令とに大きく二つに分かれる。命令形態#F1～#F4において、load

命令の場合は図6Aに示すように命令にコード「0」が記述され、add命令の場合は命令にコード「1」が記述され、cmp命令の場合は命令にコード「2」が各々記述される。

【0072】

命令形態#F5はjump命令であって、この場合は命令にコード「3」が記述される。cmp命令に関しては比較結果が同じであった場合は、図3に示したラッチ回路510のフラグ状態信号S23に基づいてzero flagが「1」にセットされ、同じでなかった場合は「0」がセットされる。

【0073】

この例では、load、add及びcmp命令はアクセスするレジスタriの番号が例えば、5ビットで表現できるときは5ビット、それ以外は13ビットで表現される。つまり、使用頻度が高い第0番のレジスタr0から第31番のレジスタr31をm=5ビットで表現する。使用頻度が低い第32番のレジスタr32から第8191番のレジスタr8191をm=13ビットで表現するようになる。

【0074】

命令形態#F1では命令長が32ビットであり、「レジスタ番号1」で示されるレジスタriの命令ビット数はn=13ビットであり、「レジスタ番号2」で示されるレジスタriの命令ビット数もn=13ビットである。命令形態#F2では命令長が24ビットであり、「レジスタ番号1」で示されるレジスタriの命令ビット数はm=5ビットであり、「レジスタ番号2」で示されるレジスタriの命令ビット数はn=13ビットである。

【0075】

また、命令形態#F3でも命令長が24ビットであり、「レジスタ番号1」で示されるレジスタriの命令ビット数はn=13ビットであり、「レジスタ番号2」で示されるレジスタriの命令ビット数はm=5ビットである。命令形態#F4は命令長が16ビットであり、「レジスタ番号1」で示されるレジスタriの命令ビット数はm=5ビットであり、「レジスタ番号2」で示されるレジスタriの命令ビット数もm=5ビットである。

【0076】

いずれの命令形態 #F1～#F4においても、最初の2ビットは命令の種類を示している。命令種類に関して、loadは転送、addは加算、cmpは比較、jumpは制御移行（分岐）を各々示している。load、add、cmp命令の場合は、命令の後に2ビットのアクセス方法#1、#2が続く。Operandは左がアクセス方法#1、「レジスタ番号1」で表され、右がアクセス方法#2、「レジスタ番号2」で表される。

【0077】

つまり、アクセス方法#1は「レジスタ番号1」で示されるレジスタriのアクセス方法を示しており、アクセス方法#2は「レジスタ番号2」で示されるレジスタriのアクセス方法を示している。アクセス方法#1とアクセス方法#2はそれぞれレジスタ番号No. 1、レジスタ番号No. 2に対応し、これらの中で処理が行われる。いずれも、図6Bに示すようにアクセス方法#1、#2には2種類が準備されている。

【0078】

アクセス方法#1、#2に関してコード「0」が記述された場合は、レジスタ番号2で示されるレジスタriに対し直接アクセスする方法である。当該レジスタ番号で示されるレジスタriの値を直接用いることを示している。アクセス方法#1、#2に関してコード「1」が記述された場合は、「レジスタ番号1」で示されるレジスタriの値をアドレスとし、当該マイクロプロセッサ101で外部メモリ2に対してアクセスする方法である（図6B参照）。

【0079】

また、図5A～図5Dにおいて、アクセス方法#1、#2の後には2ビットの「レジスタ種類1」、「レジスタ種類2」が続けて記述されている。「レジスタ種類1」は「レジスタ番号1」で示されるレジスタriの種類を示し、「レジスタ種類2」は「レジスタ番号2」で示されるレジスタriの種類を示している。レジスタ種類は図6Cに示すように2種類が準備されている。「レジスタ種類1」及び「レジスタ種類2」に関して、コード「0」が記述される場合は、レジスタ番号が「31」以下で使用頻度が高いレジスタriを示している。このレジス

タ r_i ($i = 0 \sim 31$) はレジスタ番号を $m = 5$ ビットで表現することができる。

【0080】

この「レジスタ種類1」及び「レジスタ種類2」に関して、「1」が記述される場合は、レジスタ番号が「32」以上で使用頻度が低いレジスタ r_i を示している。このレジスタ r_i ($i = 32 \sim 8191$) はレジスタ番号を $n = 13$ ビットで表現するようになされる。このように、レジスタ番号の大きさを区別することでプログラムを圧縮することができる。

【0081】

この「レジスタ種類1」、「レジスタ種類2」の後には「レジスタ番号1」、「レジスタ番号2」が続けて記述されている。「レジスタ番号1」は例えば、被数を保持するレジスタ r_i を示し、「レジスタ番号2」は加数を保持するレジスタ r_i を示す。

【0082】

また、図5Eに示す `jump` 命令のフォーマットによれば、最初の2bitに命令が記述され、続く2bitにはフラグ状態 (`flag condition`) が記述される。続く20bitにはジャンプアドレスが記述される。フラグ状態は図6Dに示すように、命令実行制御を移すかどうかの判断をするための条件である。コード「0」は「無条件」で常に制御を移す。コード「1」は「`zero flag`」で `zero flag` が「1」である場合に、制御を移す。コード「2」は「`non-zero flag`」で `zero flag` が「0」である場合に制御を移すようになされる。コード「3」は未使用である。

【0083】

続いて、プログラム作成系Iにおける処理例について説明をする。図7はプログラム作成系Iにおけるプログラム作成例を示す表図である。図7において、P1はプログラム記述画面をイメージし、P2にはその記述内容を示し、P3には実施例で該当する条件を記述したものである。所定のプログラム言語に基づいて目的の演算処理を実行するための命令を、図1に示したプログラム作成装置200で編集して圧縮プログラムを作成するためである。

【0084】

プログラム作成装置200では図1に示した表示装置24に図7に示すC言語によるプログラム記述画面P1を表示して、キーボード22及びマウス23を使用して圧縮プログラムが作成される。このとき、データベース21からマイクロプロセッサ101のプログラム作成に必要なデータが読み出される。例えば、C言語によるプログラムの記述に必要な「Global変数宣言」、「関数宣言」、「Local変数宣言」、「代入」、「加算」、「比較」及び「分岐」が読み出される。

【0085】

この例で新規な設計製造に係るマイクロプロセッサ101で、 $N=8192$ 個の32ビットのレジスタ r_i を使用する場合であって、8192個のレジスタ r_i に第0番から第8191番のシリアル番号を付与したとき、第0番から第31番のグループのレジスタ $r_0 \sim r_{31}$ を使用頻度が高い部類として「Local変数宣言」がなされる。つまり、この例では第31番目以下のレジスタ r_i に「Local変数宣言」が割り当てられる。また、第32番から第8191番のグループのレジスタ $r_{32} \sim r_{8191}$ を使用頻度が低い部類として「Global変数宣言」がなされる。つまり、第32番目以上のレジスタ r_i には「Global変数宣言」が割り当てられる。

【0086】

プログラム作成装置200では「Global変数宣言」がなされた第32番から第8191番のグループのレジスタ $r_{32} \sim r_{8191}$ の命令ビット数を $n=13$ ビットとしたとき、Local変数宣言がなされたレジスタ $r_0 \sim r_{31}$ の命令ビット数は、これよりも8ビット少ない $m=5$ ビットに圧縮される。これと共に、当該プログラムの命令構造の中に「レジスタ種類1」、「レジスタ種類2」を記述して命令長の異なる圧縮プログラムAPを作成するようになされる。

【0087】

図1に示した制御装置25ではレジスタ r_i の使用頻度に応じて命令の長さを可変するようになされる。この例では、使用頻度が高いレジスタ $r_0 \sim r_{31}$ を短い命令ビット数 $m=5$ ビットで命令セットし、使用頻度が低いレジスタ r_{32}

～r 8 1 9 1は長い命令ビット数 $n=13$ ビットで命令セットするようになされる。このようにすると、頻繁にアクセスするレジスタr 0～r 3 1は短い長さの命令をセットすることができ、マイクロプロセッサ1 0 1に実装されるプログラム格納用のROM 1 4のメモリ容量を削減することができる。

【0088】

続いて、プログラム作成装置200におけるコンパイル例について説明をする。図8はプログラム作成装置200におけるコンパイル例を示すフローチャート（メインルーチン）である。図9はコンパイラにおける代入及び演算処理例を示すフローチャート（サブルーチン）である。

【0089】

この実施例ではプログラム作成系IでC言語によるプログラムに基づいて目的の演算処理を実行するための命令を編集して圧縮プログラムを作成する場合を前提とする。また、マイクロプロセッサ101が8192個のレジスタr 0～r 8 1 9 1を使用する場合であって、8192個のレジスタr iに第0番から第8191番のシリアル番号が付与される場合を例にとる。

【0090】

これを処理条件にして、プログラム作成系IではC言語によるプログラムに基づいて命令を編集するために、図8にフローチャートのステップC1でプログラムアドレスを「0」にする。その後、ステップC2に移行してC言語によるプログラムを一行読み込む。このとき、表示装置24のプログラム記述画面P1には、例えば、「global変数宣言」を示す

```
int *read add *write add , counter, end val;
```

が表示され、また、関数宣言を示す

```
void main ()
```

```
{
```

が表示される。

【0091】

そして、ステップC3で当該プログラムの記述が「global変数宣言」であるかをチェックする。当該記述が「global変数宣言」である場合は、ス

テップC4に移行して第32番目以上のレジスタ $r_{32} \sim r_{8191}$ を割り当てる。このグループのレジスタ $r_{32} \sim r_{8191}$ を指定する命令ビット数を m ビットとしたとき、 $m=13$ ビットである。命令は命令形態#F1で作成される。その後、ステップC14に移行する。

【0092】

また、ステップC3で当該記述が「global変数宣言」ではない場合はステップC5に移行して「local変数宣言」かをチェックする。このとき、表示装置24のプログラム記述画面P1には、例えば、「local変数宣言」を示す

```
int    temp, added val;
```

が表示される。当該記述が「local変数宣言」である場合は、ステップC6に移行して第31番目以下のレジスタ $r_0 \sim r_{31}$ を割り当てる。レジスタ r_i を使用する頻度が高いことから、当該レジスタ r_i を指定する命令ビット数 n を、「global変数宣言」されたレジスタ $r_{32} \sim r_{8191}$ よりも8ビット少ない5ビットに圧縮するためである。命令は命令形態#F2乃至#F4で作成される。その後、ステップC14に移行する。

【0093】

上述のステップC5で当該記述が「local変数宣言」ではない場合はステップC7に移行して、C言語のプログラムで代入・加算処理等の実行を示す「do」が記述されているかをチェックする。このとき、表示装置24のプログラム記述画面P1には、例えば、「do」を示す

```
do {
    temp = *read add;
    temp = temp + added val;
    *write add = temp
    read add = read add + added val;
    write add = write add + added val;
    counter = counter + added val;
}
```

が表示される。このような代入・加算等の処理を示す「do」が記述されている場合は、ステップC8に移行して現在のプログラムアドレスを記憶する。その後、ステップC14に移行する。

【0094】

ステップC7で「do」が記述されていない場合はステップC9に移行してC言語のプログラムでその間の処理を示す「while」が記述されているかをチェックする。このとき、表示装置24のプログラム記述画面P1には、例えば、「while」を示す

```
while (counter !=end val) ;
```

が表示される。このような比較・分岐等の処理を示す「while」が記述されている場合は、ステップC10に移行して代入・演算処理を実行する。

【0095】

例えば、図9に示すサブルーチンをコールして、そのフローチャートのステップE1でC言語のプログラムにおいて、当該行が「while」を記述している行かがチェックされる。「while」が記述されている行の場合は、ステップE2に移行して後続の処理で生成する命令をcmp命令とする。その後、ステップE6に移行する。

【0096】

ステップE1で「while」が記述されていない行の場合は、ステップE3に移行して演算処理は加算かをチェックする。演算処理が加算の場合はステップE4に移行して後続の処理で生成する命令をadd命令とする。演算処理が加算ではない場合はステップE5に移行して後続の処理で生成する命令をload命令とする。その後、ステップE6に移行する。

【0097】

ステップE6ではレジスタriに書き込まれる変数に対応するレジスタ番号と、レジスタアレイ11から読み出される変数に対応するレジスタriのレジスタ番号を調べられる。書込みアドレスAw及び読み出しアドレスArを決めるためである。その後、ステップE7に移行する。

【0098】

ステップE 7では両方のレジスタ番号が「3 2」以上かをチェックする。両方のレジスタ番号が「3 2」以上の場合はステップE 8に移行して図5 Aに示した命令形態# F 1で命令を生成する。この命令形態# F 1で「レジスタ種類1」及び「レジスタ種類2」には「1」が記述される。このとき、「レジスタ種類1」及び「レジスタ種類2」は圧縮プログラムの命令構造の中に記述される。例えば、第3 2番から第8 1 9 1番のグループのレジスタ $r_{32} \sim r_{8191}$ に関して「レジスタ種類1」及び「レジスタ種類2」に「1」が記述される。その後、図8に示したメインルーチンのステップC 1 0にリターンする。

【0099】

また、ステップE 7で両方のレジスタ番号が「3 2」以上ではない場合はステップE 9に移行して両方のレジスタ番号が「3 1」以下かをチェックする。ここで両方のレジスタ番号が「3 1」以下の場合はステップE 1 0に移行して図5 Dに示した命令形態# F 4で命令を生成する。この命令形態# F 4で「レジスタ種類1」及び「レジスタ種類2」には「0」が記述される。このとき、「レジスタ種類1」及び「レジスタ種類2」は圧縮プログラムの命令構造の中に記述される。例えば、第0番から第3 1番のグループのレジスタ $r_0 \sim r_{31}$ に関して「レジスタ種類1」及び「レジスタ種類2」に「0」が記述される。その後、図8に示したメインルーチンのステップC 1 0にリターンする。

【0100】

更に、両方のレジスタ番号が「3 1」以下ではない場合はステップE 1 1に移行してレジスタアレイ1 1から読み出される変数のレジスタ r_i の番号が「3 2」以上かをチェックする。読み出される変数のレジスタ r_i の番号が「3 2」以上の場合は、ステップE 1 2に移行して図5 Bに示した命令形態# F 2で命令を生成する。この命令形態# F 2で「レジスタ種類1」に「0」が記述され、「レジスタ種類2」には「1」が記述される。その後、図8に示したメインルーチンのステップC 1 0にリターンする。

【0101】

更にまた、レジスタアレイ1 1から読み出される変数のレジスタ r_i の番号が「3 2」以上ではない場合は、ステップE 1 3に移行して図5 Cに示した命令形

態 # F 3 で命令を生成する。この命令形態 # F 3 では「レジスタ種類 1」に「1」が記述され、「レジスタ種類 2」に「0」が記述される。その後、図 8 に示したメインルーチンのステップ C 1 0 にリターンする。その後、ステップ C 1 1 に移行して j u m p 命令を生成する。j u m p 命令の飛び先は先に記憶したプログラムアドレスを用いる。その後、ステップ C 1 4 に移行する。

【0102】

上述のステップ C 9 で「w h i l e」が記述されていない場合はステップ C 1 2 に移行して C 言語のプログラムにおいて、データの代入又は加算かをチェックする。データの代入又は加算の場合はステップ C 1 3 に移行してデータの代入又は演算処理を実行する。このステップ C 1 3 では、図 9 に示したサブルーチンをコールして、そのフローチャートのステップ E 1 ～E 1 3 を経て図 8 に示したメインルーチンのステップ C 1 3 にリターンする。その後、ステップ C 1 4 に移行する。

【0103】

また、ステップ C 1 2 で C 言語のプログラムにおいて、データの代入又は加算ではない場合はステップ C 1 4 に移行する。ステップ C 1 4 では C 言語のプログラムに関して最後の行かをチェックされる。最後の行ではない場合は、ステップ C 1 5 に移行してプログラムアドレスを進める。その後、ステップ C 2 に戻って上述したコンパイル処理を繰り返すようになされる。最後の行に至ってこのコンパイル処理を終了する。

【0104】

これにより、図 5 A ～図 5 E に示したような命令形態 # F 1 ～# F 5 であって、命令長の異なる圧縮プログラム A P を作成することができる。この圧縮プログラム A P において、第 0 番から第 3 1 番のグループのレジスタ r 0 ～r 3 1 を指定する命令ビット数に関しては $m = 5$ ビットであり、第 3 2 番から第 8 1 9 1 番のグループのレジスタ r 3 2 ～r 8 1 9 1 を指定する命令ビット数に関しては $n = 13$ ビットである。

【0105】

続いて、プログラム実行系 II における処理例について説明をする。図 1 0 は復

元された演算プログラムによる演算命令の例を示す表図である。図11はレジスタ $r_0, r_1 \dots r_{32}, r_{33}, r_{34}, r_{35}$ 等の状態例、図12は外部メモリ2におけるデータ格納例を各々示すイメージ図である。

【0106】

この例では、データ処理装置100に接続された外部メモリ2の中に図12に示すような10個のメモリセルの配列を二組用意する。一方はメモリ配列#M1で、他方はメモリ配列#M2である。そして、図10に示す8つの演算命令 (Instruction) #I1～#I8に基づいて、その一組のメモリ配列#M1に格納された値に「1」を加算し、もう一組のメモリ配列#M2にその結果を格納する演算処理の例を挙げる。

【0107】

図10に示す演算命令 (Instruction) #I1～#I8はROM14の圧縮プログラムAPを復元した後の演算プログラムによるものである。この演算プログラムでは図11に示すように、アクセス頻度が高いレジスタ r_i が二つあるため、これらをそれぞれ r_0 と r_1 に割り当てた。これにより、プログラム全体の長さを圧縮する前の演算プログラムに比べて効率よく短縮することができた。

【0108】

また、図11において、レジスタ番号 r_0 で示されるレジスタは一時的に使用され、レジスタ番号 r_1 で示されるレジスタには加算値「1」が格納される。また、レジスタ番号 r_{32} で示されるレジスタには読み出しアドレス「0」が格納され、レジスタ番号 r_{33} で示されるレジスタには書込みアドレス「10」が格納され、レジスタ番号 r_{34} で示されるレジスタにはカウンタの初期値「0」が格納され、レジスタ番号 r_{35} で示されるレジスタには演算回数値 (終了値) 「10」が格納される。

【0109】

図10に示す各々の演算命令 #I1～#I8には、ニーモニックによる表現、機械語による表現及び処理の内容が示されている。演算命令 #I1は図5に示した命令構造において、機械語で140020hによって表される load, r_0 , (r_{32}) であり、レジスタアレイ11のレジスタ r_{32} の値をアドレスとし

、外部メモリ2から読み出した値をレジスタr0に格納する内容である。動作としては例えば、レジスタr32の値を「0」としたとき、図12に示した外部メモリ2の読み出しアドレスが「0」の内容である、メモリ配列#M1のデータ「0」が読み出され、このデータ「0」がレジスタr0に書き込まれる。

【0110】

演算命令#I2は機械語で4001hによって表されるadd, r0, r1であり、レジスタアレイ11のレジスタr0にレジスタr1の値を加算し、その結果をレジスタr0に格納する内容である。動作としてはレジスタr0の内容である「0」にレジスタr1の値である「1」が加算され、その結果「1」がレジスタr0に書き込まれる。

【0111】

演算命令#I3は機械語で280420hによって表されるload, (r33), r0であり、レジスタアレイ11のレジスタr33の値をアドレスとして、レジスタr0の値を外部メモリ2に書き込む内容である。動作としてはレジスタr33が示す外部メモリ2のメモリ配列#M2のアドレスにデータ「1」が書き込まれる。

【0112】

演算命令#I4は機械語で480401hによって表されるadd, r32, r1であり、レジスタアレイ11のレジスタr32にレジスタr1の値を加算して、その結果をレジスタr32に格納する内容である。動作としてはレジスタr32の内容である「0」にレジスタr1の値である「1」が加算され、その結果「1」がレジスタr32に書き込まれる。

【0113】

演算命令#I5は機械語で480421hによって表されるadd, r33, r1であり、レジスタアレイ11のレジスタr33にレジスタr1の値を加算して、その結果をレジスタr33に格納する内容である。動作としてはレジスタr33の内容である「0」にレジスタr1の値である「1」が加算され、その結果「1」がレジスタr33に書き込まれる。

【0114】

演算命令#I6は機械語で480441hによって表されるadd, r34, r1であり、レジスタアレイ11のレジスタr34にレジスタr1の値を加算して、その結果をレジスタr34に格納する内容である。動作としてはレジスタr34の内容である「0」にレジスタr1の値である「1」が加算され、その結果「1」がレジスタr34に書き込まれる。この演算命令#I4～#I6によって実行ステートマシーン51内のカウンタでは読み出しアドレスAr及び書き込みアドレスAwに関して「1」が加算される。

【0115】

演算命令#I7は機械語で8C044023hによって表されるcmp, r34, r35であり、レジスタアレイ11のレジスタr34の内容とレジスタr35の内容とを比較し、その値が同じ場合は、zero flagに「1」をセットし、異なっている場合は「0」にセットする内容である。動作としてはレジスタr34とレジスタr35の値である「1」と「10」は異なるので、zero flagには「0」がセットされる。zero flagの値はラッチ回路510によって保持され、以降の命令によって参照される。

【0116】

演算命令#I8は機械語でE00000hによって表されるjump nz, LOOPであり、zero flagが「0」の場合は、LOOPで示されるラベルへ制御を移す内容である。動作としては、zero flagが「0」の場合は制御を演算命令#I1に移す。上記の動作が10回、繰り返されるとレジスタr34の値が「10」になり、演算命令#I7によりzero flagが「1」にセットされ、演算命令#I8で制御が移らなくなり、演算処理を終了するようになされる。このように、全てのレジスタriの命令ビット数を単一の方法で表現した場合と比べて、効率良くROM14を使用することが可能になる。

【0117】

続いて、マイクロプロセッサ101における動作例について説明をする。図13はマイクロプロセッサ101における動作例を示すフローチャートである。図14及び図15は命令ビット復元デコーダ13における処理例（その1, 2）を示すフローチャートである。

【0118】

この実施例ではマイクロプロセッサ101がプログラム実行系IIを構成し、ROM14から読み出された圧縮プログラムAPから図10に示した演算命令#I1～#I8を含む演算プログラムを復元する。このとき、命令形態#F1～#F4に関して「レジスタ種類1」及び、「レジスタ種類2」にコード「0」が記述されている場合は、命令ビット復元デコーダ13によってレジスタ番号の拡張が行われる。この際の命令ビットの拡張では、例えば、「レジスタ番号1」の命令ビット数 $m=5$ ビットの上位に8ビットの「0」を追加するようになされる。この演算プログラムに基づいて、図12に示した外部メモリ2の中のメモリ配列#M1の値に「1」を加算し、メモリ配列#M2に格納するようになされる。

【0119】

レジスタアレイ11のレジスタ状態については、図11に示したように例えば、6個のレジスタ r_0 , r_1 , r_{32} , r_{33} , r_{34} , r_{35} に関して、 r_0 が不定、 r_1 が初期値「1」、 r_{32} 及び r_{34} が共に初期値「0」、 r_{33} 及び r_{35} が初期値「10」が設定される。これらの値を書き込む場合は、実行ステートマシーン51ではレジスタアレイ11に書き込みアドレスAwが出力され、その初期値「0」、「1」、「10」が設定される。

【0120】

これを動作条件にして、図13に示すフローチャートのステップF1で、まず、命令ビット復元デコーダ13はROM14から圧縮プログラム（機械語命令）APを順次受け取り、このプログラムAPを解読して所定の命令長の演算命令#I1～#I8を検出する。

【0121】

この命令ビット復元デコーダ13は例えば、図14に示すサブルーチンをコールしてそのフローチャートのステップG1で命令部分を取り出し、命令信号S9を実行ステートマシーン51に出力する。これと共に、命令ビット復元デコーダ13ではステップG2に移行して当該命令形態がjump命令であるかをチェックする。当該命令形態が#F5で示されるjump命令の場合はステップG12に移行してflag condition、jump addressを出力す

る。その後、メインルーチンのステップF1にリターンする。

【0122】

また、ステップG2で当該命令形態がjump命令でない場合は、ステップG4に移行して当該命令形態に関して「レジスタ種類1」に記述されているコードは「0」又は「1」かをチェックする。「レジスタ種類1」にコード「0」が記述されている場合はステップG4に移行して「レジスタ番号1」の命令ビット数mを5bit長として圧縮プログラムAPから取り出す。その後、ステップG5に移行して「レジスタ番号1」の命令ビット数m=5bitの上位8ビットに「0」を付加して13bit長とする。その後、ステップG7に移行する。

【0123】

上述のステップG3で「レジスタ種類1」にコード「1」が記述されている場合はステップG6に移行して「レジスタ番号1」の命令ビット数nを13bit長として圧縮プログラムAPから取り出す。その後、ステップG7に移行して、当該命令形態に関して「レジスタ種類2」に記述されているコードは「0」又は「1」かをチェックする。「レジスタ種類2」にコード「0」が記述されている場合はステップG8に移行して「レジスタ番号2」の命令ビット数mを5bit長として圧縮プログラムAPから取り出す。その後、ステップG9に移行して「レジスタ番号2」の命令ビット数m=5bitの上位8ビットに「0」を付加して13bit長とする。その後、ステップG11に移行する。

【0124】

上述のステップG7で「レジスタ種類2」にコード「1」が記述されている場合はステップG10に移行して「レジスタ番号2」の命令ビット数nを13bit長として圧縮プログラムAPから取り出す。その後、ステップG11に移行して、「レジスタ番号1」、「レジスタ番号2」、「アクセス方法#1」及び「アクセス方法#2」を検出する。

【0125】

その後、図13に示したメインルーチンのステップF1にリターンする。従って、実行ステートマシン51には「レジスタ種類1」及び「レジスタ種類2」は出力されず、所定の命令長の演算命令#I1～#I8に基づく命令制御信号S4

、命令信号 S 9 及び各引数信号 S 1 0 が出力される。

【 0 1 2 6 】

この命令信号 S 9 には l o a d 命令、 a d d 命令、 c m p 命令、 j u m p 命令が含まれる。各引数信号 S 1 0 にはアクセス方法 # 1, アクセス方法 # 2, レジスタ番号 r 0, r 1 . . . 等、フラグ状態 (f l a g c o n d i t i o n) 及びジャンプアドレス等が含まれる。命令制御信号 S 4 はデコーダ 1 3 から命令読出しステートマシーン 5 2 に出力される。

【 0 1 2 7 】

なお、ROM 1 4 で圧縮プログラム A P を読み出す場所 (アドレス) はプログラムカウンタ 5 4 (P C) によって指定される。これらの読み出しの動作は命令読み出しステートマシーン 5 2 によって制御される。

【 0 1 2 8 】

命令読出しステートマシーン 5 2 では命令ビット復元デコーダ 1 3 から出力される命令制御信号 S 4 に基づいてプログラムカウンタ 5 4 及び実行ステートマシーン 5 1 を制御する。例えば、当該マシーン 5 2 は命令ビット復元デコーダ 1 3 から命令信号 S 9 及び各引数信号 S 1 0 が実行ステートマシーン 5 1 へ出力されると共に命令実行開始信号 S 2 9 を出力する。

【 0 1 2 9 】

実行ステートマシーン 5 1 では命令実行開始信号 S 2 9 に基づいて命令の実行を開始する。例えば、データの書込み時には、書込み制御信号 S w がレジスタアレイ 1 1 に出力され、セクタ 5 9 には選択制御信号 S 2 4 が出力される。データの読出し時には、レジスタアレイ 1 1 に読出しアドレス A r が出力される。

【 0 1 3 0 】

演算時には、ラッチ制御信号 S 3 4 がラッチ回路 5 8 に出力され、ラッチ回路 5 1 0 にはラッチ制御信号 S 3 8 が出力される。当該プロセッサ外部には外部制御信号 S 1 6 が出力される。命令の実行が終了すると、実行ステートマシーン 5 1 は命令読み出しステートマシーン 5 2 へ実行終了信号 S 2 6 を出力し、プログラムカウンタ 5 4 の値を進めるようになされる。

【 0 1 3 1 】

プログラムカウンタ54ではカウント制御信号S30に基づいてROM14から圧縮プログラムAPを読み出す場所が指定される。+1インクリメンタ55はプログラムカウンタ54のカウント出力信号S5を「+1」してインクリメントするようになされる。

【0132】

そして、ステップF2で実行ステートマシーン51は命令読出しステートマシーン52の命令読出し制御を受けて演算命令#I1を受け取ると、機械語で140020hによって表されるload, r0, (r32)に基づいて書込み信号S16を外部メモリ2に出力する。この値はレジスタアレイ11の読み出しアドレスArとして用いられる。レジスタアレイ11は32番目の値をデータ信号線L20に出力する。

【0133】

この値はラッチ回路511によって保持され、アドレスバス19Bを経由し、外部メモリ2へ出力される。そして、レジスタアレイ11のレジスタr32の値をアドレスとし、外部メモリ2から読み出した値をレジスタr0に格納する。このとき、レジスタr32の値が「0」であるので、図12に示した外部メモリ2のアドレスが「0」の内容であるメモリ配列#M1のデータ「0」が読み出され、このデータ「0」がレジスタr0に書き込まれる。

【0134】

つまり、外部メモリ2からアドレスバス19Bによって転送されたアドレス（場所）のデータがセクタ59に出力される。実行ステートマシーン51ではこのデータが選択されるように、選択信号S24を出力する。これにより、データがレジスタアレイ11に入力される。そして、実行ステートマシーン51では書き込みアドレスAw=「0」をレジスタアレイ11に出力する。その後、実行ステートマシン51は書き込み信号Swを用いて実際に、演算結果値の書き込みを指示するようになされる。

【0135】

その後、ステップF3で実行ステートマシン51は命令読出しステートマシーン52の命令読出し制御を受けて演算命令#I2を受け取ると、機械語で400

1 hによって表される `add, r0, r1` に基づいてレジスタアレイ 11 のレジスタ `r0` にレジスタ `r1` の値を加算し、その結果をレジスタ `r0` に格納する。このとき、レジスタ `r0` の内容である「0」にレジスタ `r1` の値である「1」が加算され、その結果「1」がレジスタ `r0` に書き込まれる。

【0136】

そして、ステップ F4 で実行ステートマシン 51 は命令読出しステートマシン 52 の命令読出し制御を受けて演算命令 # I3 を受け取ると、機械語で 280420 h によって表される `load, (r33), r0` に基づいてレジスタアレイ 11 のレジスタ `r33` の値をアドレスとして、レジスタ `r0` の値を外部メモリ 2 に書き込む。このとき、レジスタ `r33` が示す外部メモリ 2 のメモリ配列 # M2 のアドレスにデータ「1」が書き込まれる。

【0137】

その後、ステップ F5 で実行ステートマシン 51 は命令読出しステートマシン 52 の命令読出し制御を受けて演算命令 # I4 を受け取ると、機械語で 480401 h によって表される `add, r32, r1` に基づいてレジスタアレイ 11 のレジスタ `r32` にレジスタ `r1` の値を加算して、その結果をレジスタ `r32` に格納する。このとき、レジスタ `r32` の内容である「0」にレジスタ `r1` の値である「1」が加算され、その結果「1」がレジスタ `r32` に書き込まれる。この演算命令 # I4 によって実行ステートマシン 51 内のカウンタでは読み出しアドレス `Ar` 及び書き込みアドレス `Aw` に関して「1」が加算される。

【0138】

そして、ステップ F6 で実行ステートマシン 51 は命令読出しステートマシン 52 の命令読出し制御を受けて演算命令 # I5 を受け取ると、機械語で 480421 h によって表される `add, r33, r1` に基づいてレジスタアレイ 11 のレジスタ `r33` にレジスタ `r1` の値を加算して、その結果をレジスタ `r33` に格納する。このとき、レジスタ `r33` の内容である「0」にレジスタ `r1` の値である「1」が加算され、その結果「1」がレジスタ `r33` に書き込まれる。この演算命令 # I5 によって実行ステートマシン 51 内のカウンタでは読み出しアドレス `Ar` 及び書き込みアドレス `Aw` に関して「1」が加算される。

【0139】

その後、ステップF7で実行ステートマシン51は命令読出しステートマシン52の命令読出し制御を受けて演算命令#I6を受け取ると、機械語で480441hによって表されるadd, r34, r1に基づいてレジスタアレイ11のレジスタr34にレジスタr1の値を加算して、その結果をレジスタr34に格納する。このとき、レジスタr34の内容である「0」にレジスタr1の値である「1」が加算され、その結果「1」がレジスタr34に書き込まれる。この演算命令#I6によって実行ステートマシン51内のカウンタでは読み出しアドレスAr及び書き込みアドレスAwに関して「1」が加算される。

【0140】

この例では、ステップF8でレジスタr35が示す値=10回に至ったかが判別される。例えば、実行ステートマシン51は命令読出しステートマシン52の命令読出し制御を受けて演算命令#I7を受け取ると、機械語で8C044023hによって表されるcmp, r34, r35に基づいてレジスタアレイ11のレジスタr34の内容とレジスタr35の内容とを比較し、その値が同じ場合は、zero flagに「1」をセットし、異なっている場合は「0」にセットする。このとき、レジスタr34とレジスタr35の値である「1」と「10」は異なるので、zero flagには「0」がセットされる。zero flagの値はラッチ回路510によって保持され、以降の命令によって参照される。

【0141】

そして、実行ステートマシン51は命令読出しステートマシン52の命令読出し制御を受けて演算命令#I8を受け取ると、機械語でE00000hによって表されるjump nz, LOOPに基づいてzero flagが「0」の場合は、LOOPで示されるラベルへ制御を移行する。zero flagが「0」の場合は制御をステップF2の演算命令#I1に移す。上記の動作がステップF8で10回繰り返されるとレジスタr34の値が「10」になり、演算命令#I7によりzero flagが「1」にセットされ、演算命令#I8で制御が移らなくなり、演算処理を終了するようになされる。

【0142】

このように、本発明に係る実施例としてのマイクロプロセッサ101によれば、8192個のレジスタ $r_0 \sim r_{8191}$ の中でその使用頻度に基づいて当該レジスタ r_i を指定する命令ビット数が予め圧縮されると共に、当該プログラムの命令構造の中に「レジスタ種類1」及び「レジスタ種類2」が記述された命令長の異なる圧縮プログラムAPに基づいてデータを処理するようになされる。

【0143】

ROM14には8192個のレジスタ $r_0 \sim r_{8191}$ の中から当該レジスタ r_i を指定するための圧縮プログラムAPが記憶される。命令ビット復元デコーダ13では、このROM14から圧縮プログラムAPを読み出して「レジスタ種類1」及び「レジスタ種類2」が解読され、この「レジスタ種類1」及び「レジスタ種類2」に基づいて当該レジスタ r_i を指定するための命令ビット数が復元される。

【0144】

従って、レジスタ r_i の使用頻度に応じて可変された命令の長さの圧縮プログラムAPであって、頻繁にアクセスするレジスタ $r_0 \sim r_{31}$ には短い長さの命令がセットされた、圧縮プログラムデータをROM14に格納することができるので、そのメモリ容量を低減することができる。この例ではプログラムを圧縮する前に比べて16ビット×32個×命令数分だけメモリ容量を低減できる。

【0145】

これにより、メモリセルや論理演算素子から成るPLDによりマイクロプロセッサ101を構築する場合に、ROMとして機能させるメモリセルの占有率を低減することができ、その分のメモリセルをレジスタに多く割り当てることができるようになる。

【0146】

【発明の効果】

以上説明したように、本発明に係るデータ処理システムによれば、一方で、所定のプログラム言語に基づいて目的の演算処理を実行するための命令を編集してプログラムを作成し、他方で、当該プログラムと複数のレジスタとを使用してデ

ータを処理する場合に、プログラム作成系で作成された圧縮プログラムを取得してレジスタ種類を解読し、このレジスタ種類に基づいて当該レジスタを指定する命令ビット数を復元し、所定長さの命令に基づいて複数のレジスタを指定して任意の演算を実行するデータ処理装置を備えるものである。

【0147】

この構成によって、プログラム作成系ではレジスタの使用頻度に応じて命令の長さを可変できるので、頻繁にアクセスするレジスタに短い長さの命令をセットすることができる。従って、プログラム実行系ではプログラムデータを格納するROM等のメモリ容量を低減することができる。また、メモリセルや論理演算素子から成るPLDによりプロセッサを構築する場合に、ROMとして機能させるメモリセルの占有率を低減することができ、その分のメモリセルをレジスタに多く割り当てることができる。

【0148】

本発明に係るデータ処理装置によれば、レジスタを使用する頻度に基づいて当該レジスタを指定する命令ビット数が予め圧縮されると共に、当該プログラムの命令構造の中にレジスタ種類が記述された命令長の異なる圧縮プログラムに基づいてデータを処理する場合に、この圧縮プログラムから解読したレジスタ種類に基づいて当該レジスタを指定するための命令ビット数を復元する命令解読復元手段を備え、この命令解読復元手段によって復元された所定長の命令に基づいてレジスタを指定して任意の演算を実行するものである。

【0149】

この構成によって、レジスタの使用頻度に応じて可変された命令の長さの圧縮プログラムであって、頻繁にアクセスするレジスタには短い長さの命令がセットされた、圧縮プログラムデータを記憶手段に格納することができるので、そのメモリ容量を低減することができる。従って、メモリセルや論理演算素子から成るPLDによりプロセッサを構築する場合に、ROMとして機能させるメモリセルの占有率を低減することができ、その分のメモリセルをレジスタに多く割り当てることができる。

【0150】

本発明に係るデータ処理方法によれば、プログラム作成系で所定のプログラム言語に基づいて目的の演算処理を実行するための命令を編集してプログラムを作成し、プログラム実行系で当該プログラムと複数のレジスタとを使用してデータを処理する場合に、プログラム作成系では、レジスタを使用する頻度に基づいて当該レジスタを指定する命令ビット数を圧縮すると共に、当該プログラムの命令構造の中にレジスタ種類を記述して命令長の異なる圧縮プログラムを作成し、プログラム実行系では、プログラム作成系で作成された圧縮プログラムを取得してレジスタ種類を解読し、ここで解読されたレジスタ種類に基づいて当該レジスタを指定する命令ビット数を復元し、ここで復元された所定長さの命令に基づいて複数のレジスタを指定して任意の演算を実行するようになされる。

【0151】

この構成によって、プログラム作成系ではレジスタの使用頻度に応じて命令の長さを可変できるので、頻繁にアクセスするレジスタに短い長さの命令をセットすることができる。従って、プログラム実行系ではプログラムデータを格納するROM等のメモリ容量を低減することができる。また、メモリセルや論理演算素子から成るPLDによりプロセッサを構築する場合に、ROMとして機能させるメモリセルの占有率を低減することができ、その分のメモリセルをレジスタに多く割り当てることができる。

【0152】

この発明は命令実行プログラム等に基づいて各種データ処理をするCPUや、MPU、PLD等、これらの組み込み電子機器に適用して極めて好適である。

【図面の簡単な説明】

【図1】

本発明に係る実施形態としてのデータ処理システム10の構成例を示すブロック図である。

【図2】

データ処理システム10における処理例を示すフローチャートである。

【図3】

本発明に係る実施例としてのマイクロプロセッサ101の構成例を示すブロッ

ク図である。

【図 4】

レジスタアレイ 1 1 の内部構成例を示すブロック図である。

【図 5】

A ～ E は ROM 1 4 にセットされる命令の構造例を示すフォーマットである。

【図 6】

A ～ D は命令構造における記述内容例を示す表図である。

【図 7】

プログラム作成系 I におけるプログラム作成例を示す表図である。

【図 8】

プログラム作成装置 2 0 0 におけるコンパイル例を示すフローチャート（メインルーチン）である。

【図 9】

コンパイラにおける代入及び演算処理例を示すフローチャート（サブルーチン）である。

【図 1 0】

復元された演算プログラムによる演算命令の例を示す表図である。

【図 1 1】

r 0 , r 1 r 3 2 , r 3 3 , r 3 4 , r 3 5 等のレジスタの状態例を示すイメージ図である。

【図 1 2】

外部メモリ 2 におけるデータ格納例を示すイメージ図である。

【図 1 3】

マイクロプロセッサ 1 0 1 における動作例を示すフローチャートである。

【図 1 4】

命令ビット復元デコーダ 1 3 における処理例（その 1）を示すフローチャートである。

【図 1 5】

命令ビット復元デコーダ 1 3 における処理例（その 2）を示すフローチャート

である。

【符号の説明】

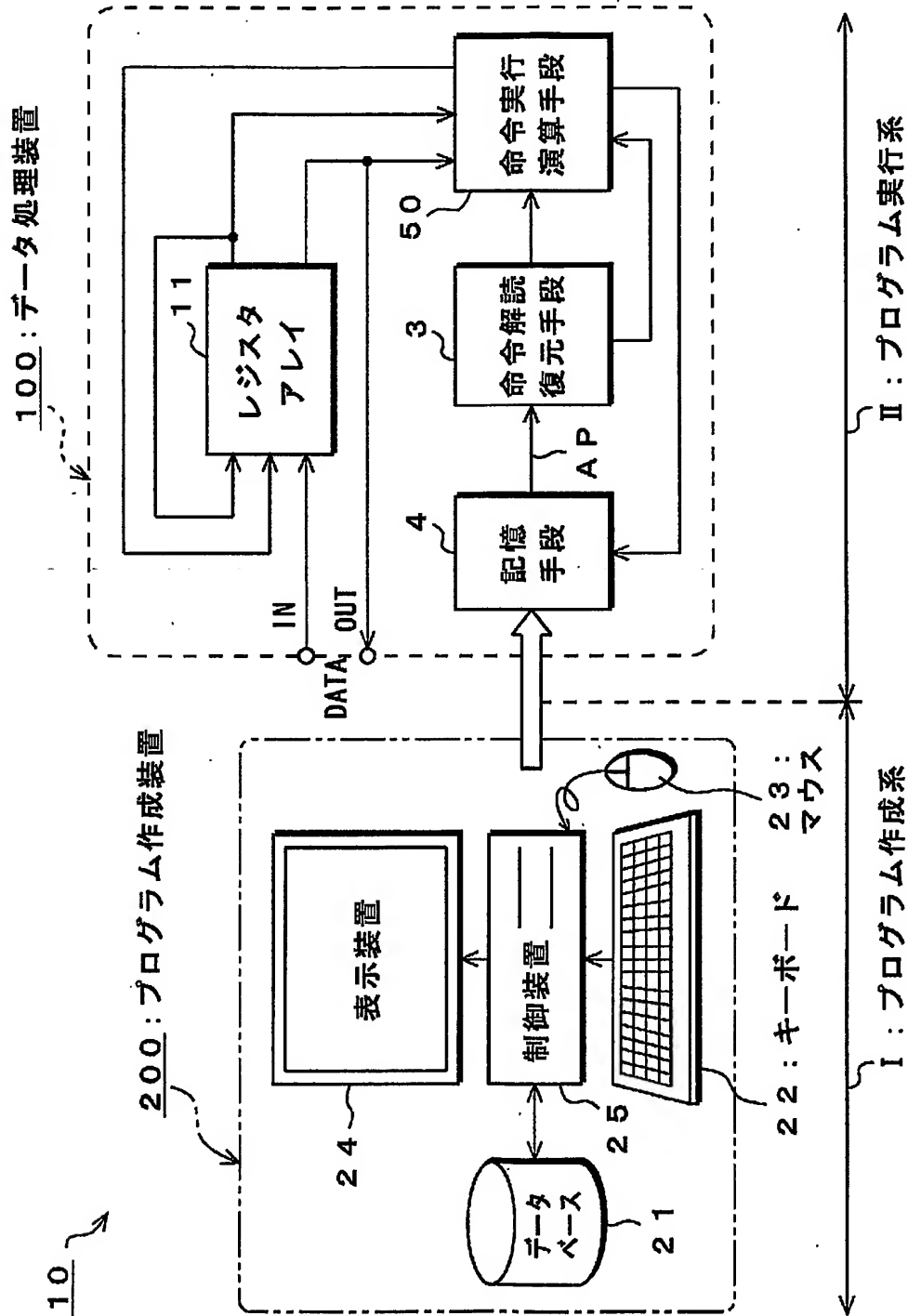
2・・・外部メモリ、3・・・命令解読復元手段、4・・・記憶手段、10・・・データ処理システム、11・・・レジスタアレイ、12・・・ALU、13・・・命令ビット復元デコーダ（命令解読復元手段）、14・・・ROM（記憶手段）、50・・・命令実行演算手段、100・・・データ処理装置、101・・・マイクロプロセッサ、200・・・プログラム作成装置

【書類名】

図面

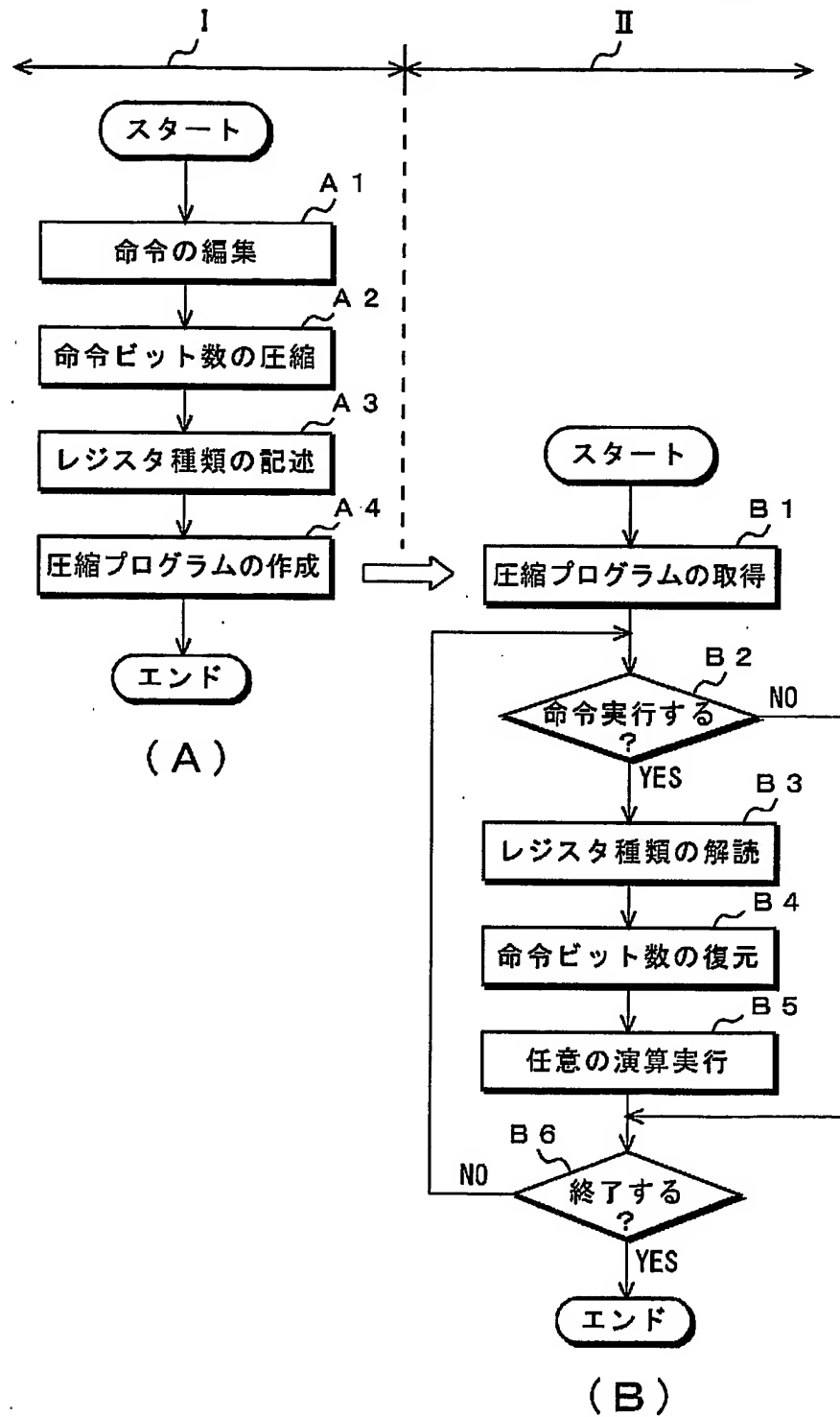
【図1】

実施形態としてのデータ処理システム10の構成例



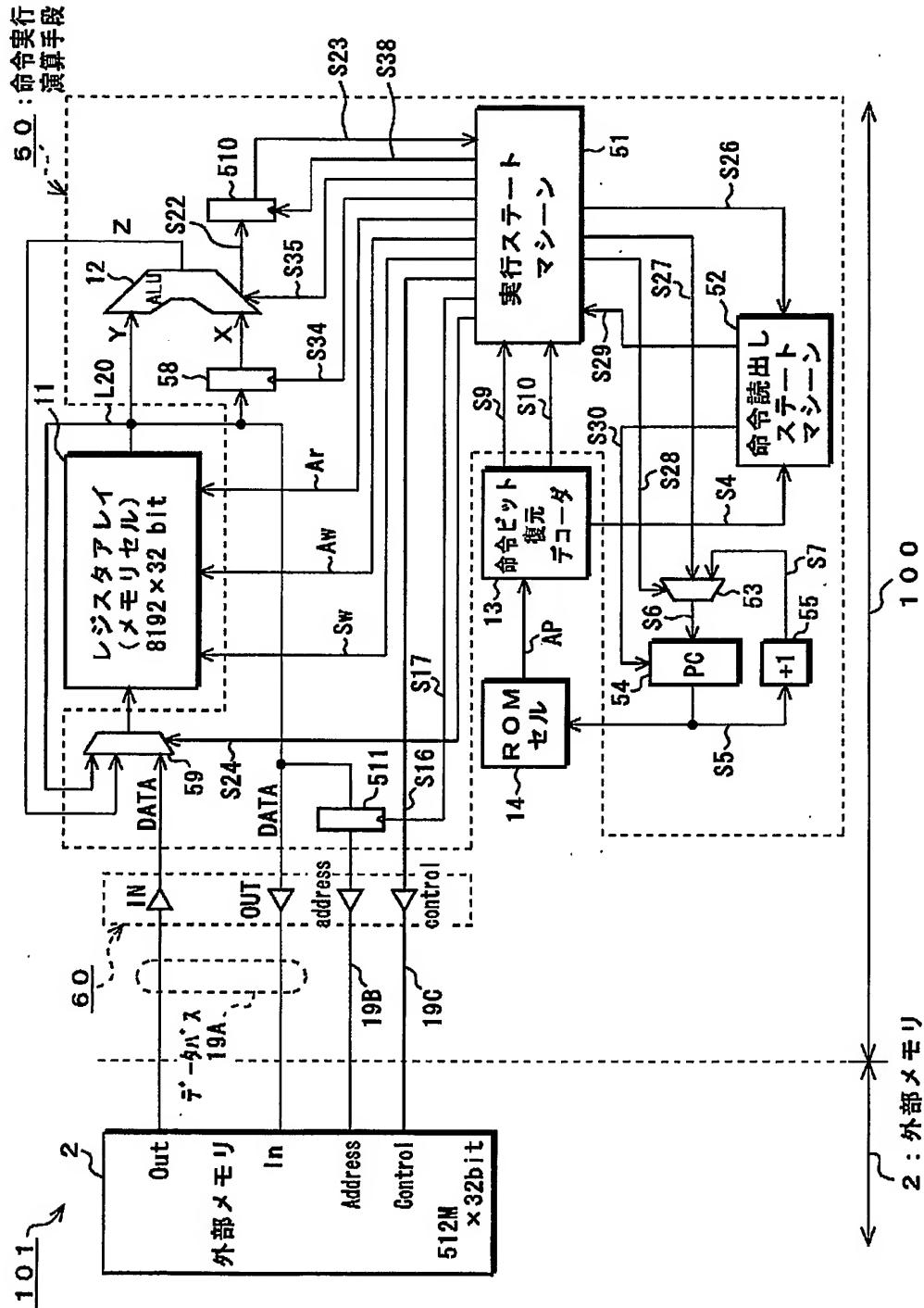
【図2】

データ処理システム10における処理例



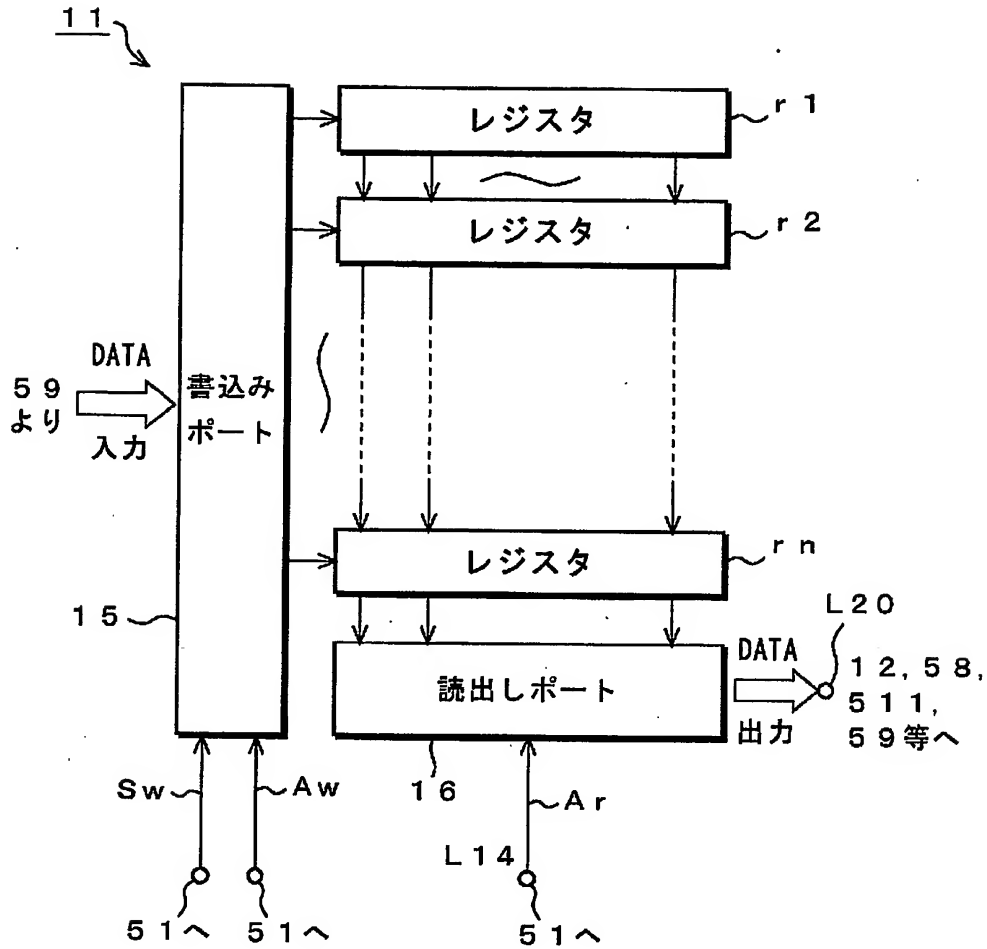
【図3】

実施例としてのマイクロプロセッサ101の構成例



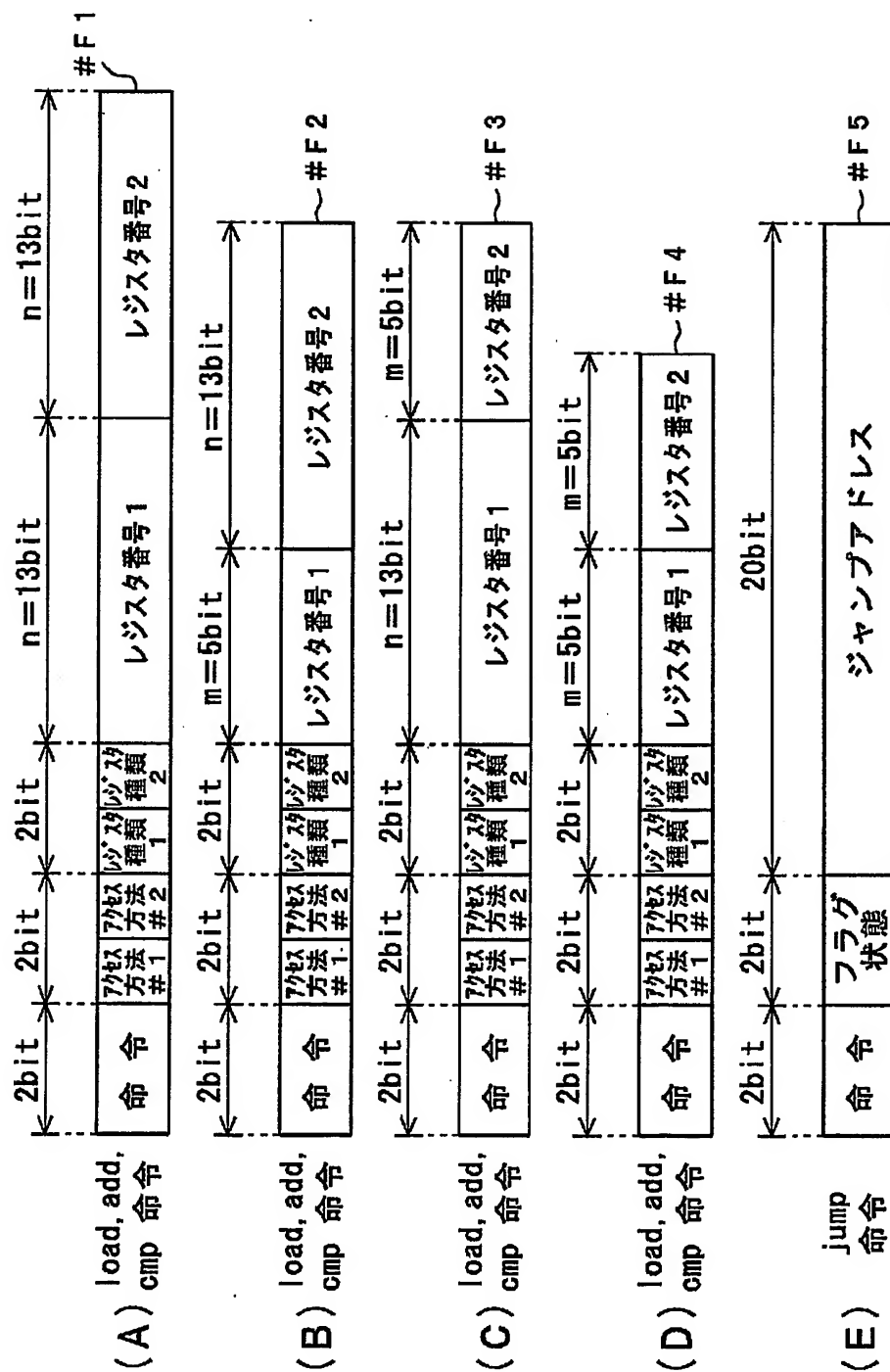
【図4】

レジスタアレイ 11 の内部構成例



【図5】

ROM 14 にセットされる命令の構造例



【図 6】

命令構造における記述内容例

命 令	
0	load
1	add
2	cmp
3	jump

(A)

アクセス方法 # 1, # 2	
0	レジスタ直接
1	レジスタ相対

(B)

レジスタ種類 1, 2	
0	レジスタ番号が 31 以下
1	レジスタ番号が 32 以上

(C)

フラグ状態	
0	無条件
1	zero flag
2	non-zero flag
3	未使用

(D)

【図 7】

プログラム作成系 I におけるプログラム作成例

C 言語によるプログラム	記述内容	実施例
<pre> int *read_add, *write_add, counter, end_val; void main() { int temp, added_val; do { temp = *read_add; temp = temp + added_val; *write_add = temp; read_add = read_add + added_val; write_add = write_add + added_val; counter = counter + added_val; } while(counter != end_val); </pre>	<p>Global変数宣言</p> <p>関数宣言</p> <p>Local変数宣言</p> <p>代入 加算 代入 加算 加算 加算</p> <p>比較、分岐</p>	<p>32 番目以上のレジスタに割り当てられる</p> <p>31 番目以下のレジスタに割り当てられる</p>

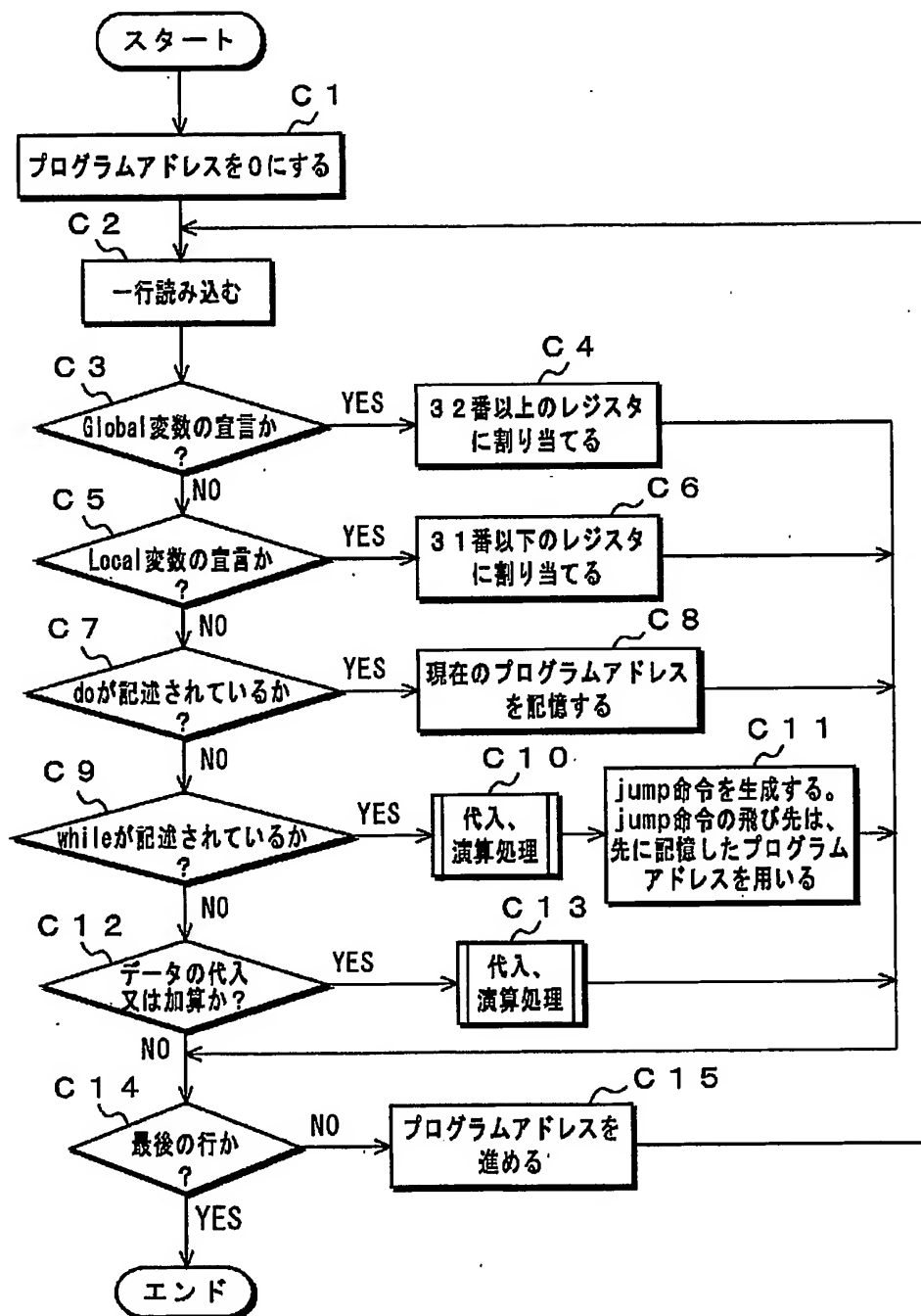
P3

P2

P1

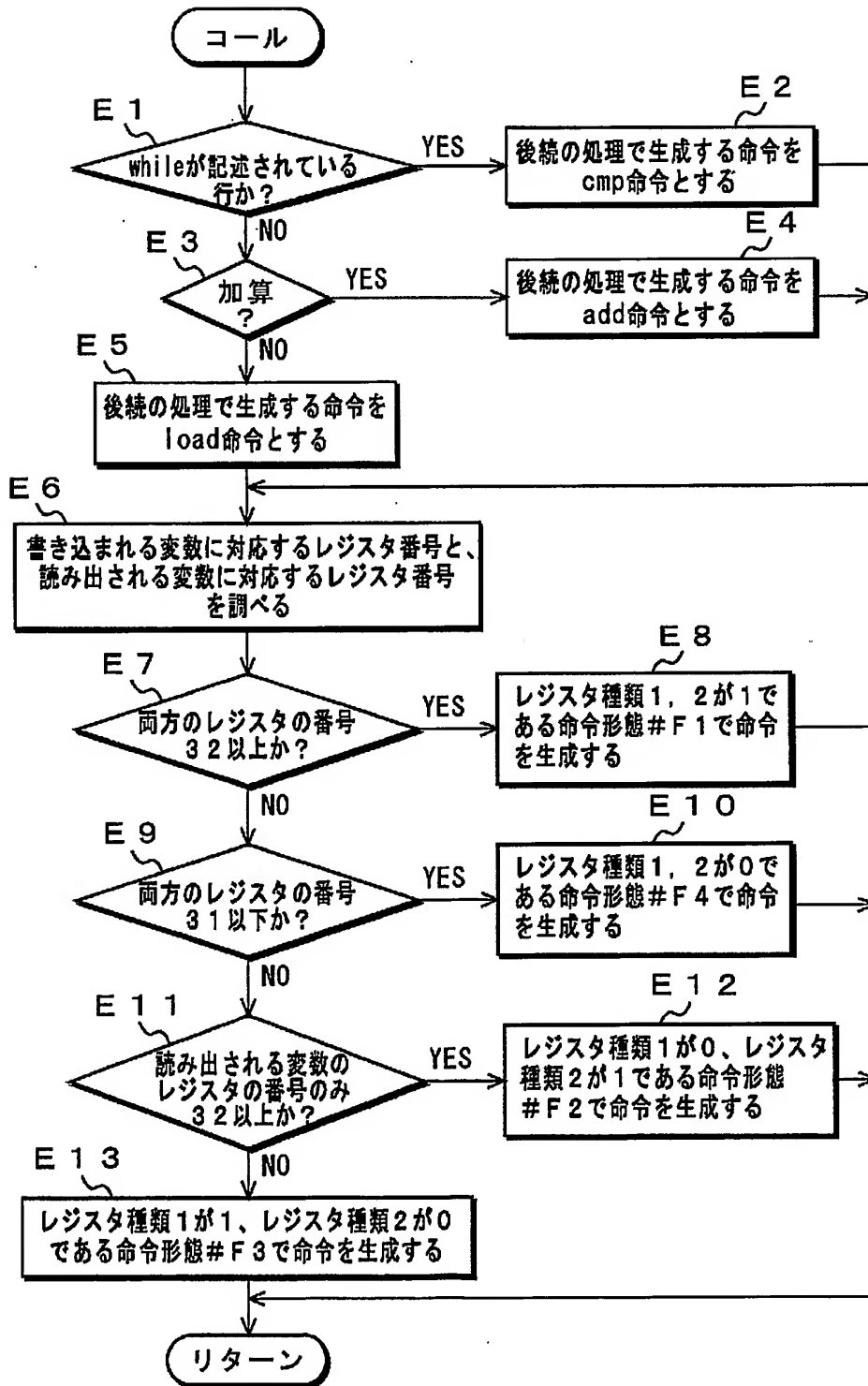
【図 8】

プログラム作成装置 200 におけるコンパイル例



【図9】

代入及び演算処理例



【図10】

演算プログラムによる演算命令の例

No.	ニーモニックによる表現		機械語による表現
# I 1	LOOP	load r 0, (r 32)	140020h
# I 2		add r 0, r 1	4001h
# I 3		load(r 33), r 0	280420h
# I 4		add r 32, r 1	480401h
# I 5		add r 33, r 1	480421h
# I 6		add r 34, r 1	480441h
# I 7		cmp r 34, r 35	8C044023h
# I 8		jump nz, LOOP	E00000h

【図11】

レジスタの状態例

レジスタ番号	値	
r 0	—	← 一時的に使用されるレジスタ
r 1	1	← 加算値
r 32	0	← 読み出しアドレス
r 33	10	← 書き込みアドレス
r 34	0	← カウンタ
r 35	10	← 終了値

【図12】

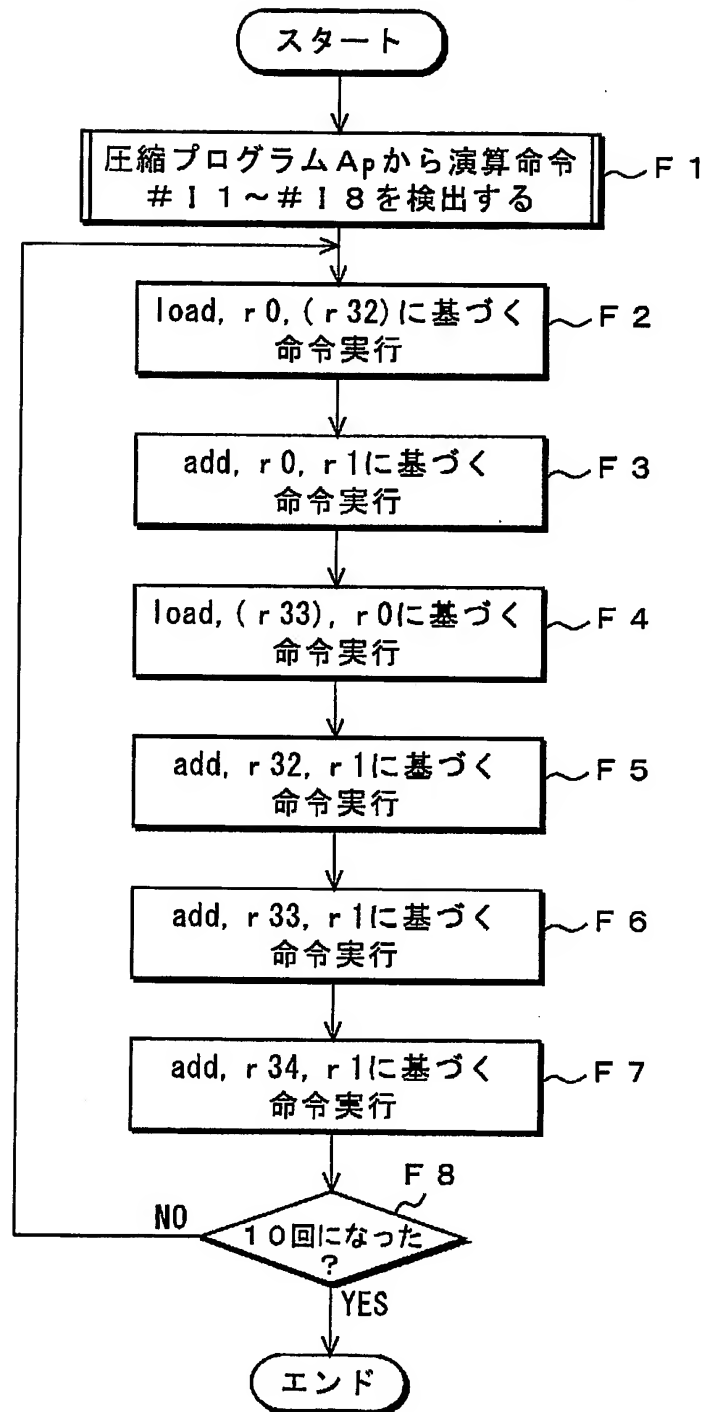
外部メモリ2におけるデータ格納例

2
↓

アドレス	データ	
0000h	0	↑ #M1 ↓
0001h	0	
0002h	0	
0003h	0	
0004h	0	
0005h	0	
0006h	0	
0007h	0	
0008h	0	
0009h	0	
000Ah	0	↑ #M2 ↓
000Bh	0	
000Ch	0	
000Dh	0	
000Eh	0	
000Fh	0	
0010h	0	
0011h	0	
0012h	0	
0013h	0	

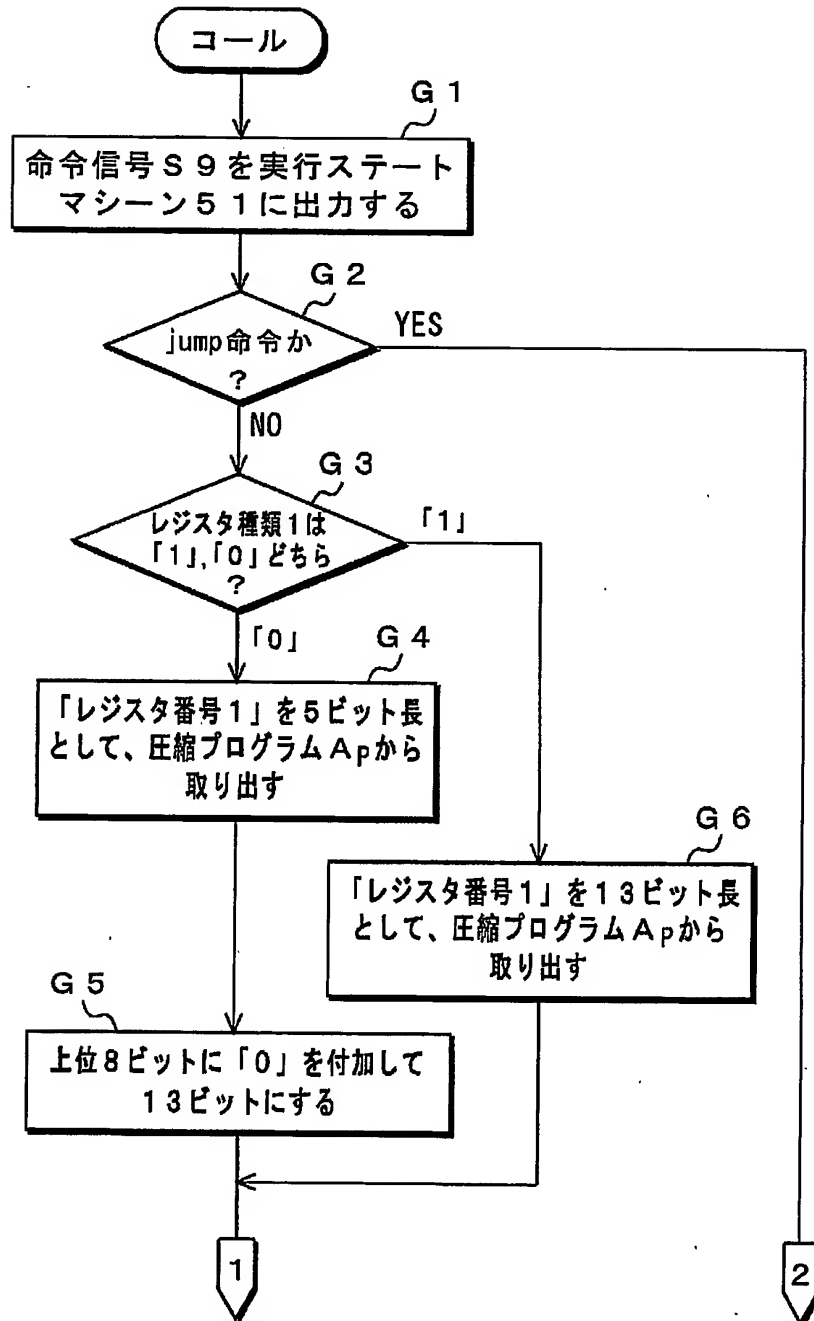
【図13】

マイクロプロセッサ101における動作例



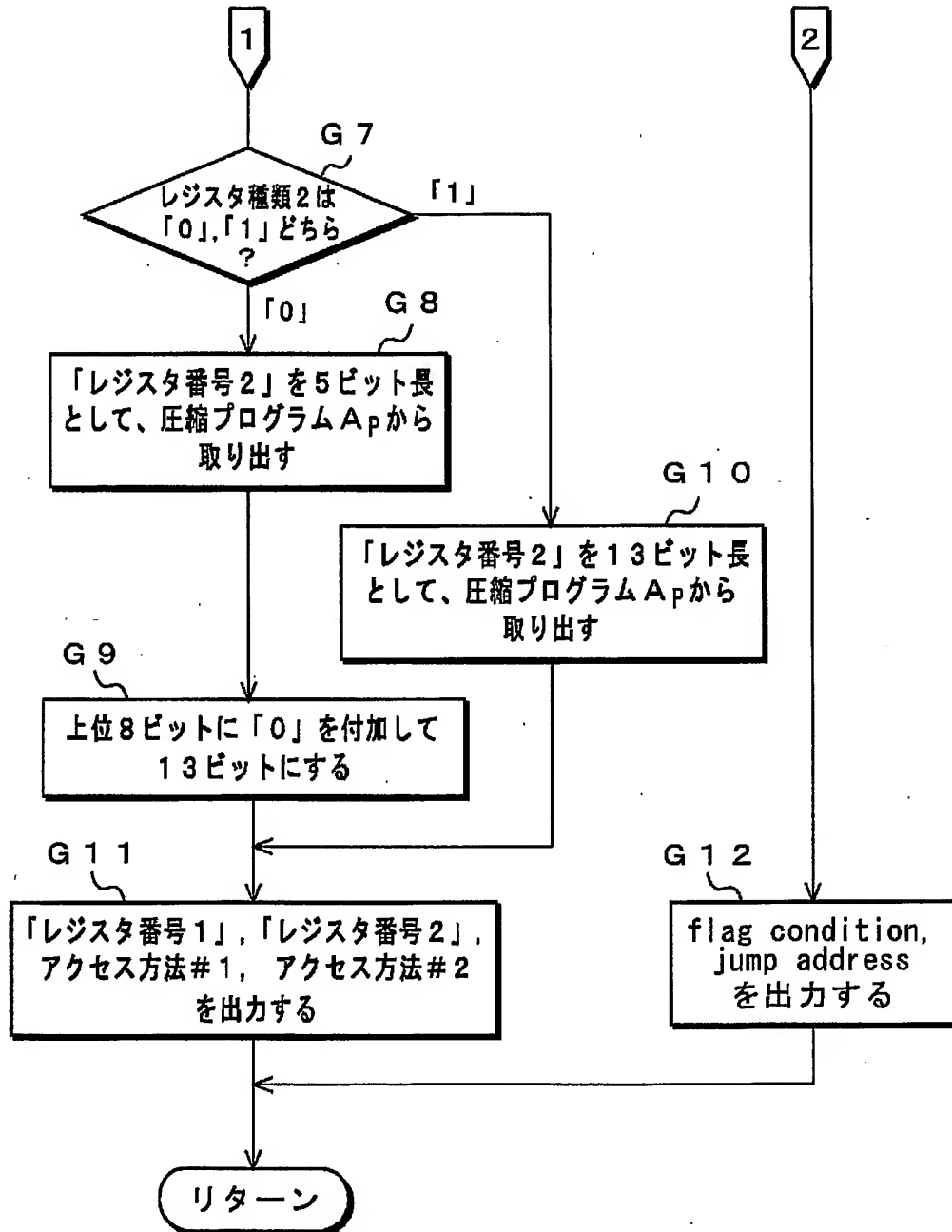
【図14】

命令ビット復元デコーダ13における処理例（その1）



【図15】

命令ビット復元デコーダ13における処理例（その2）



【書類名】 要約書

【要約】

【課題】 レジスタの使用頻度に応じて命令の長さを可変できるようにすると共に、頻繁にアクセスするレジスタには短い長さの命令をセットできるようにし、プログラムデータを格納するROM等のメモリ容量を低減できるようにする。

【解決手段】 レジスタ r_i を使用する頻度に基づいて当該レジスタ r_i を指定する命令ビット数を圧縮すると共に、当該プログラムの命令構造の中にレジスタ種類を記述して命令長の異なる圧縮プログラムAPを作成するプログラム作成装置200と、このプログラム作成装置200で作成された圧縮プログラムAPを取得してレジスタ種類を解読し、レジスタ種類に基づいて当該レジスタ r_i を指定する命令ビット数を復元し、所定の命令長の命令構造に復元したプログラムに基づいて複数のレジスタ r_i を指定して任意の演算を実行するデータ処理装置100とを備えるものである。

【選択図】 図1

出 願 人 履 歴 情 報

識別番号 [000002185]

1. 変更年月日 1990年 8月30日
[変更理由] 新規登録
住 所 東京都品川区北品川6丁目7番35号
氏 名 ソニー株式会社

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☒ FADED TEXT OR DRAWING
- ☒ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.